

Diameter Maintenance and Extensions (DIME)
Internet-Draft
Intended Status: Proposed Standard
Expires: August 11, 2014

B. Hirschman
L. Bertz
Sprint
February 2014

Diameter Congestion and Filter Attributes
draft-bertz-dime-congestion-flow-attributes-02.txt

Abstract

This document defines optional ECN and filter related attributes that can be used for improved traffic identification, support of ECN and minimized filter administration within Diameter.

RFC 5777 defines a Filter-Rule AVP that accommodates extensions for classification, conditions and actions. It does not support traffic identification for packets using Explicit Congestion Notification as defined in RFC 3168 and does not provide specific actions when the flow(s) described by the Filter-Rule are congested.

A Filter-Rule can describe multiple flows but not the exact number of flows. Flow count and other associated data (e.g. packets) is not captured in Accounting applications, leaving administrators without useful information regarding the effectiveness or understanding of the filter definition.

These optional attributes are forward and backwards compatible with RFC 5777.

Status of This Memo

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

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

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

This Internet-Draft will expire on February 14, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	4
2. Terminology and Abbreviations	4
3. ECN-IP-Codepoint, Congestion-Treatment and Filter Attributes .	4
3.1. ECN-IP-Codepoint AVP	4
3.2. Congestion-Treatment AVP	5
3.3. Flow-Count AVP	5
3.4. Packet-Count AVP	5
4. IANA Considerations	5
4.1. AVP Codes	5
5. Security Considerations	6
6. Acknowledgements	6
7. References	6
7.1. Normative References	6
Authors' Addresses	6

1. Introduction

Two optional Explicit Congestion Notification (ECN) [RFC3168] related AVPs are specified in the document. The first AVP provides direct support for ECN [RFC3168] in the IP header and the second AVP provides the ability to define alternate traffic treatment when congestion is experienced.

This document also defines two optional AVPs, Flow-Count and Packet-Count, used for conveying flow information within the Diameter protocol [RFC6733]. These AVPs were found to be useful for a wide range of applications. The AVPs provide a way to convey information of the group of flows described by the Filter-Rule, IPFilterRule or other Diameter traffic filters.

The semantics and encoding of all AVPs can be found in Section 3.

Such AVPs are, for example, needed by some ECN applications to determine the number of flows congested or used by administrators to determine the impact of filter definitions.

Additional parameters may be defined in future documents as the need arises. All parameters are defined as Diameter-encoded Attribute Value Pairs (AVPs), which are described using a modified version of the Augmented Backus-Naur Form (ABNF), see [RFC6733]. The data types are also taken from [RFC6733].

2. Terminology and Abbreviations

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

3. ECN-IP-Codepoint, Congestion-Treatment and Filter Attributes

3.1. ECN-IP-Codepoint AVP

The ECN-IP-Codepoint AVP (AVP Code TBD) is of type Enumerated and specifies the Explicit Congestion Notification codepoint values to match in the IP header.

Value	Binary	Keyword	References
0	00	Non-ECT (Not ECN-Capable Transport)	[RFC3168]
1	01	ECT(1) (ECN-Capable Transport)	[RFC3168]
2	10	ECT(0) (ECN-Capable Transport)	[RFC3168]
3	11	CE (Congestion Experienced)	[RFC3168]

When this AVP is used for classification in the Filter-Rule it MUST be part of Classifier Grouped AVP as defined in RFC5777.

3.2. Congestion-Treatment AVP

The Congestion-Treatment AVP (AVP Code TBD) is of type Grouped and indicates how congested traffic, i.e., traffic that has Explicit Congestion Notification Congestion Experienced marking set or some other administratively defined criteria, is treated. In case the Congestion-Treatment AVP is absent the treatment of the congested traffic is left to the discretion of the node performing QoS treatment.

```

Congestion-Treatment ::= < AVP Header: TBD >
    { Treatment-Action }
    [ QoS-Profile-Template ]
    [ QoS-Parameters ]
    * [ AVP ]

```

Treatment-Action, QoS-Profile-Template and QoS-Parameters are defined in [RFC5777]. The Congestion-Treatment AVP is an action and MUST be an attribute of the Filter-Rule Grouped AVP as defined in RFC5777.

3.3. Flow-Count AVP

The Flow-Count AVP (AVP Code TBD) is of type Unsigned64.

It indicates the number of protocol specific flows. The protocol is determined by the filter (e.g. IPFilterRule, Filter-Id, etc.).

3.4. Packet-Count AVP

The Packet-Count AVP (AVP Code TBD) is of type Unsigned64.

It indicates the number of protocol specific packets. The protocol is determined by the filter (e.g. IPFilterRule, Filter-Id, etc.).

4. IANA Considerations

4.1. AVP Codes

IANA allocated AVP codes in the IANA-controlled namespace registry specified in Section 11.1.1 of [RFC6733] for the following AVPs that are defined in this document.

+-----+-----+-----+-----+-----+-----+					
AVP		Code	Defined	Data Type	Section

ECN-IP-Codepoint	TBD 3.1	Enumerated
Congestion-Treatment	TBD 3.2	Grouped
Flow-Count	TBD 3.3	Unsigned64
Packet-Count	TBD 3.4	Unsigned64

5. Security Considerations

The document does not raise any new security concerns. This document describes an extension of RFC5777 that introduces a new filter parameter applied to ECN as defined by [RFC3168]. It also defines a new Grouped AVP that expresses what action to take should congestion be detected. The Grouped AVP reuses attributes defined in RFC5777.

The security considerations of the Diameter protocol itself have been discussed in RFC 6733 [RFC6733]. Use of the AVPs defined in this document MUST take into consideration the security issues and requirements of the Diameter base protocol.

6. Acknowledgements

We would like to thank Avi Lior for his guidance and feedback during the development of this specification.

7. References

7.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3168] Black, D., Floyd, S., and K. Ramakrishnan, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001.
- [RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.
- [RFC5777] Korhonen, J., Tschofenig, H., Arumaithurai, M., Lior, A. and Jones, M. Ed., "Traffic Classification and Quality of Service (QoS) Attributes for Diameter", RFC 5777, February 2010.

Authors' Addresses

Lyle Bertz
Sprint

6220 Sprint Parkway
Overland Park, KS 66251
United States

EMail: Lyle.T.Bertz@sprint.com

Brent Hirschman
Sprint
6220 Sprint Parkway
Overland Park, KS 66251
United States

EMail: Brent.Hirschman@sprint.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: August 22, 2013

B. Campbell
Tekelec
H. Tschofenig
Nokia Siemens Networks
J. Korhonen
Renesas Mobile
A. Roach
Mozilla
February 18, 2013

Diameter Overload Data Analysis
draft-campbell-dime-overload-data-analysis-00

Abstract

When a Diameter server or agent becomes overloaded, it needs to be able to gracefully reduce its load, typically by informing clients to reduce sending traffic for some period of time. Multiple mechanisms have been proposed for transporting overload and load information. While these proposals differ in many ways, they share similar data requirements. This document analyzes the data requirements of each proposal with a view towards proposing a common set of Diameter Attribute-Value Pairs (AVPs).

Status of this Memo

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

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

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

This Internet-Draft will expire on August 22, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal

Provisions Relating to IETF Documents
(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Documentation Conventions	3
3. Overload Control Data Usage	3
4. Mechanism Differences that Affect Data Structures	4
4.1. Non-Adjacent Nodes	4
4.2. Stateless Negotiation	4
4.3. Overload Scopes	5
4.4. Hard or Soft Overload State	5
5. Naming Conventions	5
6. Data Element Comparison	6
6.1. Data Elements for Connection Establishment and Negotiation	6
6.1.1. Supported Scope Selection	6
6.1.2. Algorithm Selection	6
6.1.3. Application Selection	7
6.1.4. Frequency of Reports	7
6.1.5. Grouping	7
6.2. Data Elements for Overload and Load reporting	7
6.2.1. Scope of Report	7
6.2.2. Overload Severity	8
6.2.3. Report Algorithm	8
6.2.4. Report Expiration	8
6.2.5. Current Load	9
6.2.6. Applications covered by a Report	9
6.2.7. Report Action	9
6.2.8. Priority	10
6.2.9. Session Groups	10
6.3. Result Codes	10
7. IANA Considerations	11
8. Security Considerations	11
9. References	12
9.1. Normative References	12
9.2. Informative References	12
Appendix A. Contributors	12
Authors' Addresses	12

1. Introduction

When a Diameter [RFC6733] server or agent becomes overloaded, it needs to be able to gracefully reduce its load, typically by informing clients to reduce sending traffic for some period of time. The Diameter Overload Control Requirements [I-D.ietf-dime-overload-reqs] describe requirements for overflow control mechanisms.

At the time of this writing, there have been two proposals for Diameter overload control mechanisms. "A Mechanism for Diameter Overload Control" (MDOC) [I-D.roach-dime-overload-ctrl] defines a mechanism that piggybacks overload and load state information over existing Diameter messages. "The Diameter Overload Control Application" (DOCA) [I-D.korhonen-dime-ovl] defines a mechanism that uses a new and distinct Diameter application to communicate similar information. While there are significant differences between the two proposals, they carry similar information. Each proposal includes its own set of Diameter AVPs.

This document is intended as a framework for discussing the data requirements of the two proposals. It includes an analysis of the differences and similarities of their respective data elements, with a view towards rationalizing the AVPs from the two proposals.

The authors expect that a follow-on effort will eventually specify a common data model for reporting Diameter overload information.

This document assumes that Diameter nodes exchange overload control information via Diameter, rather than via some out-of-band channel. This document does not address the specific difference of either mechanism proposal, except where they impact the AVP definitions.

2. Documentation Conventions

This document uses terms defined in [RFC6733] and [I-D.ietf-dime-overload-reqs].

3. Overload Control Data Usage

A Diameter overload control mechanism based on the overload control requirements [I-D.ietf-dime-overload-reqs] involves the exchange of information between two or more Diameter nodes. The exchanged information serves three distinct purposes:

Negotiation: Diameter nodes need to negotiate support for the overload control mechanism in general. Nodes that support overload control need to advertise the overload control scopes they can support. Finally, they need to select an overload control algorithm.

Communication of Overload State: Nodes need to report that an overload condition is in effect, to what degree they are overloaded, and the scope of the overload condition. We refer to such a communication as an "Overload Report".

Communication of Load: Nodes need to communicate their current load status, even when not in an overloaded state.

Overload Control information may be communicated between adjacent Diameter nodes, or it may cross one or more intervening nodes. Overload Control information can be communicated in either direction; that is, a downstream node can indicate overload to an upstream node, or vice-versa.

Open Issue: There is an ongoing discussion about whether the overload control mechanism should be strictly hop-by-hop, or whether it should support communication between non-adjacent nodes. The results of this discussion may have implications for overload control data elements.

4. Mechanism Differences that Affect Data Structures

While a thorough comparison of the two proposed mechanisms is out of scope for this document, there are a few differences that directly impact the choice of data elements.

4.1. Non-Adjacent Nodes

MDOC only supports hop-by-hop communication of overload information. DOCA allows for the possibility of communication between non-adjacent nodes. For hop-by-hop communication, the originator of an overload report is always the directly connected node. If non-adjacent communication is to be allowed, the data model needs a way to express the identity of the originating node.

4.2. Stateless Negotiation

Both MDOC and DOCA allow overload control parameters to be negotiated at the beginning of a connection, and persist for the duration of the connection. DOCA also allows a "stateless" mode, where the parameters do not persist between overload reports. This requires

the sender of an overload report to restate any relevant parameters for each report. Thus, the DOCA overload report format includes the ability to express all such parameters at any time, not just during negotiation.

Note that stateless negotiation does not mean that no state may ever be saved. Nodes may use implementation-specific methods of remembering certain parameters, or out-of-band configuration methods to do the same.

4.3. Overload Scopes

As described in [I-D.ietf-dime-overload-reqs], it's possible for a Diameter node to experience overload that impacts some subset of potential traffic. For example, a Diameter agent might route traffic to different servers based on realm. If the server for one realm experienced an outage or overload condition, the agent report that it is overloaded for that realm, but can process traffic for other realms normally. We use the term "overload scope", or simply "scope", to refer to the set of potential messages affected by an overload report.

MDOC includes a richer (and therefore more complex) concept of overload scopes. A node may include multiple scopes in an overload report. Each scope entry indicates both the type of scope, and the value of the scope, where the value is interpreted according to the type.

DOCA also allows a node to include multiple scopes in a report. But DOCA's current set of scope types only affect the interpretation of the originating node identity. Therefore the DOCA scope entries do not include a value.

4.4. Hard or Soft Overload State

MDOC assumes that overload information is soft state. That is, it expires if not refreshed within a stated interval. DOCA also treats most overload information as soft state, but there are situations where it may be treated as hard-state. For example, if the OC-Level is set to "Hold", the expiration time is not honored.

5. Naming Conventions

MDOC and DOCA use somewhat different naming conventions for their respective AVPs. DOCA prefixes each AVP name with "OC". (for example, "OC-Scope"). MDOC prefixes AVPs that can appear in the root of messages with "Overload", and leaves those that occur inside an

overload related grouped AVP to be identified by context. (For example, "Overload Info" and "Supported Scopes"). The working group should consider picking one approach or the other.

6. Data Element Comparison

6.1. Data Elements for Connection Establishment and Negotiation

The following sections describe data elements used for initial negotiation.

6.1.1. Supported Scope Selection

- o DOCA: OC-Scope : Bitmap of scopes supported by the sender. Currently defined values are "Host scope", "Realm Scope", "Only origin realm", "Application Information", "Node Utilization Information", and "Application Priorities".
- o MDOC: Supported-Scopes : Bitmap of scopes supported by the sender. Currently defined values are "Destination-Realm", "Application-ID", "Destination-Host", "Host", "Connection", "Session-Group", and "Session".

DOCA uses OC-Scope both to declare supported scopes, and to list the scopes associated with a particular overload report. MDOC uses separate dedicated AVPs for the two purposes. DOCA overloads OC-Scope to include indicators that load information and priority information may be included.

6.1.2. Algorithm Selection

- o DOCA: OC-Algorithm : Bitmap of supported algorithms. Currently defined values are "Drop", "Throttle", and "Prioritize". Multiple values allowed.
- o MDOC: Overload-Algorithm: Enumeration of supported algorithms. Multiple instances allowed in negotiation. Currently, there is one algorithm described, namely "loss".

Both mechanisms support algorithm extensibility. MDOC only allows Overload-Algorithm to occur in a CER or CEA message, and negotiates a single algorithm for the duration of the connection. DOCA allows the algorithm to be selected at report time. (Open Issue: what does it mean to indicate multiple algorithms in a congestion report?)

6.1.3. Application Selection

- o DOCA: OC-Applications: Indications of the applications that are of interest.
- o MDOC: MDOC assumes that overload reports can apply to any and all applications, and does not negotiate the list upfront. The "application" scope is used to select one or more applications on a per-report basis.

Open Issue: Are there use cases for the up front negotiation of applications of interest?

6.1.4. Frequency of Reports

- o DOCA: OC-Tocl: Indicates how frequent reports shall be sent.
- o MDOC: N/A

Since MDOC piggybacks overload reports in existing messages, the rate of overload reports is the same as the overall message rate. This may have advantage of giving more rapid and precise feedback as load increases.

Open Issue: We need further discussion about the appropriate rate(s) for overload reporting, regardless of which mechanism may be selected.

6.1.5. Grouping

- o DOCA: n/a - negotiation AVPs included at message root.
- o MDOC: Load-Info: Grouped AVP acting as a container for the other AVPs used for negotiation.

6.2. Data Elements for Overload and Load reporting

6.2.1. Scope of Report

- o DOCA: OC-Scope (See Section 6.1.1)
- o MDOC: Load-Info-Scope: Octet-String giving the scope of the overload report. The string contains a type indicator and a value. One or more instances required.

MDOC has a richer and more complex concept of scopes. Multiple scopes can be combined for a given overload report. Allowable scope combinations are described in [I-D.roach-dime-overload-ctrl].

6.2.2. Overload Severity

- o DOCA: OC-Level: OctetString(1): Values 1-6 define discreet overload levels of increasing severity, with 1 meaning no overload condition, and 6 meaning clients should switch to a different server.
- o DOCA: OC-Sending-Rate: Float32: Used when the "throttle" algorithm is in effect to indicate the maximum desired Diameter message rate.
- o MDOC: Overload-Metric (Unsigned32): A numeric representation of load. The meaning is up to the interpretation of the selected algorithm, with the exception that a value of zero always means that no overload abatement is in effect. For the "Loss" algorithm, Overload Metric is a numeric value in the range of zero through 100, indicating the percentage of traffic reduction requested.

The Overload-Metric AVP used by MDOC is more general than OC-Level, in that it's interpretation is left to the algorithm. The meaning of the OC-Level values appear to be fixed regardless of algorithm choice. the OC-Level meanings could be used in MDOC by defining a new algorithm that interpreted Overload-Metric values 1-6 in the same way as defined for OC-Level.

Since MDOC does not define an algorithm similar to "throttle", it has no built in analog to OC-Sending-Rate. However, since MDOC allows algorithm extensibility, one could define a similar algorithm, and if necessary, add an extension AVP to state sending-rate.

6.2.3. Report Algorithm

- o DOCA: OC-Algorithm (See Section 6.1.2)
- o MDOC: The overload control algorithm is set during negotiation, and doesn't change for the duration of the connection.

Open Issue: DOCA's reuse of the OC-Algorithm AVP seems to allow more than one algorithm to be assigned to a single overload report. It's not clear what that would mean.

6.2.4. Report Expiration

- o DOCA: OC-Best-Before: (Time) Time of report expiration.
- o MDOC: Period-Of-Validity (Unsigned32)- Number of seconds until expiration.

DOCA defines expiration to be a point in time. MDOC uses a duration, i.e. number of seconds until expiration. The DOCA approach seems to require clock synchronization.

DOCA contains an open issue about whether to allow reports to expire vs. requiring explicit signaling.

6.2.5. Current Load

- o DOCA: OC-Utilization: Indicates the overall load situation as a value between 0 and 100.
- o MDOC: Load: The load situation in terms of 0 - 65535.

Current load indicates the existing load on an otherwise non-overloaded node. MDOC's range of 0-65535 was selected to harmonize with the DNS service location (SRV) [RFC2782] record's "Weight" field.

6.2.6. Applications covered by a Report

- o DOCA: OC-Applications: Indications what applications are of interest for load reporting.
- o MDOC does not use a separate AVP for this purpose. Rather, one or more applications can be indicated using the application scope type.

6.2.7. Report Action

- o DOCA: OC-Action: Indicates the start, interim, and end of an overload period.
- o MDOC: MDOC does not have a separate AVP to indicate the start and stop of an overload condition. Rather, a report with a non-zero Overload-Metric value starts the condition, and a report with a zero value, or the expiration of the Period-of-Validity value, indicate an end. Subsequent reports with non-zero Overload-Metric values serve the same purpose as a DOCA report with an OC-Action value of "interim".

Open Issue: Is OC-Action redundant? DOCA also has the ability to express a non-overload condition in OC-Level, so an approach similar to that of MDOC should be workable.

6.2.8. Priority

- o DOCA: OC-Priority: Unsigned32: When used in an OC-Information AVP, sets the relative priority of applications listed in OC-Applications. As specified, may also be used to set the priority of a given Diameter message. [Open Issue: Is OC-Priority only in effect when the "Prioritize" algorithm is in effect?]
- o MDOC: N/A

MDOC does not have an explicit priority data element. Relative priority between applications can be managed using the "Application" scope. This is not exactly the same as stating inter-application priority explicitly, but it may be possible to accomplish similar behavior.

6.2.9. Session Groups

- o DOCA: N/A
- o MDOC: Session-Group: UTF8String: Session-Group allows a node to assign a session to a named group. Overload Reports can refer to all sessions in a group using the Session-Group AVP.

A common application for Session-Group is when a Diameter agent load balances Diameter sessions across a set of servers. If the agent assigns all of the sessions assigned to a particular server to a group, and that server later becomes overloaded, the agent can send one overload report that applies to all sessions in the group, but does not apply to sessions assigned to other, non-overloaded, servers.

DOCA may be able to do something similar using by using the OC-Origin AVP to identify the overloaded server. However, the server-group approach can work even if the Diameter agent performs topology hiding.

6.3. Result Codes

DOCA defines the following Diameter result codes:

- o DIAMETER_NO_COMMON_SCOPE (Permanent Failure): The Diameter peers are unable to negotiate one or more scopes in common.
- o DIAMETER_NO_COMMON_ALGORITHM (Permanent Failure): The Diameter peers are unable to negotiate one or more algorithms in common.

- o `DIAMETER_TOCL_TOO_SMALL` (Permanent Failure): The peer included an OC-TOCL AVP with an unacceptably low value.
- o `DIAMETER_TOCL_TOO_BIG` (Permanent Failure): The peer included an OC-TOCL AVP with an unacceptably high value.
- o `DIAMETER_RATE_TOO_BIG` (Permanent Failure): The peer included an OC-SENDING-RATE AVP with an unacceptably high value.

A failure to negotiate Overload Control support does not cause a connection failure in MDOC. Instead, overload control is just not invoked on the connection.

MDOC defines the following result codes:

- o `DIAMETER_PEER_IN_OVERLOAD` (Transient Failure): When a Diameter node drops a request due to overload, it responds with this result code. This is primarily used when the peer does not support overload control, and therefore fails to reduce load as it would be expected to do so if it supported overload control.

`DIAMETER_PEER_IN_OVERLOAD` may be of value to both mechanisms. The Overload Control Requirements [I-D.ietf-dime-overload-reqs] argues that the result codes in the Diameter base protocol are insufficient for reporting failures due to congestion.

7. IANA Considerations

This draft makes no requests of IANA. The authors expect that a follow-on effort will specify a common set of Overload Control AVPs. This may introduce additional IANA considerations.

8. Security Considerations

This document compares the data elements used by "DOCA" [I-D.korhonen-dime-ovl] and MDOC [I-D.roach-dime-overload-ctrl]. It introduces no security considerations beyond those in the respective documents.

The authors expect that a follow-on effort will specify a common set of Overload Control AVPs. This may introduce additional security considerations.

The authors made no attempt to analyze the security considerations in the DOCA and MDOC specifications for completeness.

9. References

9.1. Normative References

- [RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.
- [I-D.ietf-dime-overload-reqs] McMurry, E. and B. Campbell, "Diameter Overload Control Requirements", draft-ietf-dime-overload-reqs-03 (work in progress), January 2013.
- [I-D.roach-dime-overload-ctrl] Roach, A., "A Mechanism for Diameter Overload Control", draft-roach-dime-overload-ctrl-01 (work in progress), October 2012.
- [I-D.korhonen-dime-ovl] Korhonen, J., "Diameter Overload Control Application", draft-korhonen-dime-ovl-00 (work in progress), October 2012.

9.2. Informative References

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

Appendix A. Contributors

Eric McMurry made significant contributions to the analysis in this draft.

Authors' Addresses

Ben Campbell
Tekelec
17210 Campbell Rd.
Suite 250
Dallas, TX 75252
US

Email: ben@nostrum.com

Hannes Tschofenig
Nokia Siemens Networks
Linnoitustie 6
Espoo 02600
Finland

Email: Hannes.Tschofenig@nsn.com

Jouni Korhonen
Renesas Mobile
Porkkalankatu 24
Helsinki FIN-00180
Finland

Email: jouni.nospam@gmail.com

Adam Roach
Mozilla
Dallas, TX

Email: adam@nostrum.com

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

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

September 23, 2014

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

Abstract

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

Status of This Memo

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

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

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

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

Copyright Notice

Copyright (c) 2014 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents

carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

This document may contain material from IETF Documents or IETF Contributions published or made publicly available before November 10, 2008. The person(s) controlling the copyright in some of this material may not have granted the IETF Trust the right to allow modifications of such material outside the IETF Standards Process. Without obtaining an adequate license from the person(s) controlling the copyright in such materials, this document may not be modified outside the IETF Standards Process, and derivative works of it may not be created outside the IETF Standards Process, except to format it for publication as an RFC or to translate it into languages other than English.

Table of Contents

1. Introduction	3
2. Terminology	4
3. Overview	4
4. Reusing Existing Diameter Applications	6
4.1. Adding a New Command	6
4.2. Deleting an Existing Command	7
4.3. Reusing Existing Commands	7
4.3.1. Adding AVPs to a Command	7
4.3.2. Deleting AVPs from a Command	9
4.3.3. Changing the Flags Setting of AVP in existing Commands	10
4.4. Reusing Existing AVPs	10
4.4.1. Setting of the AVP Flags	10
4.4.2. Reuse of AVP of Type Enumerated	11
5. Defining New Diameter Applications	11
5.1. Introduction	11
5.2. Defining New Commands	11
5.3. Use of Application-Id in a Message	12
5.4. Application-Specific Session State Machines	13
5.5. Session-Id AVP and Session Management	13
5.6. Use of Enumerated Type AVPs	14
5.7. Application-Specific Message Routing	16
5.8. Translation Agents	17
5.9. End-to-End Application Capabilities Exchange	17
5.10. Diameter Accounting Support	18
5.11. Diameter Security Mechanisms	20
6. Defining Generic Diameter Extensions	20
7. Guidelines for Registrations of Diameter Values	21

8. IANA Considerations	23
9. Security Considerations	23
10. Contributors	24
11. Acknowledgments	24
12. References	25
12.1. Normative References	25
12.2. Informative References	25
Authors' Addresses	27

1. Introduction

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

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

1. Addition of new functionality to an existing Diameter application without defining a new application.
2. Addition of new functionality to an existing Diameter application that requires the definition of a new application.
3. The definition of an entirely new Diameter application to offer functionality not supported by existing applications.
4. The definition of a new generic functionality that can be reused across different applications.

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

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

- o Discuss design choices and provide guidelines when defining new applications.
- o Present trade-off choices.

2. Terminology

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

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

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

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

3. Overview

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

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

1. Defining new AVP values
2. Creating new AVPs

3. Creating new commands

4. Creating new applications

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

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

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

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

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

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

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

4. Reusing Existing Diameter Applications

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

4.1. Adding a New Command

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

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

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

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

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

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

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

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

4.2. Deleting an Existing Command

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

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

4.3. Reusing Existing Commands

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

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

4.3.1. Adding AVPs to a Command

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

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

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

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

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

- o Would it be required for the receiving side to be able to process and understand the AVP and its content?
- o Would the new AVPs change the state machine of the application?
- o Would the presence of the new AVP lead to a different number of round-trips, effectively changing the state machine of the application?
- o Would the new AVP be used to differentiate between old and new variances of the same application whereby the two variances are not backward compatible?
- o Would the new AVP have duality in meaning, i.e., be used to carry application-related information as well as to indicate that the message is for a new application?

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

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

- o Use of optional AVPs with intersecting meaning. One AVP has partially the same usage and meaning as another AVP. The presence of both can lead to confusion.
- o An optional AVPs with dual purpose, i.e., to carry application data as well as to indicate support for one or more features. This has a tendency to introduce interpretation issues.
- o Adding one or more optional AVPs and indicating (usually within descriptive text for the command) that at least one of them has to be understood by the receiver of the command. This would be equivalent to adding a mandatory AVP, i.e., an AVP with the M-bit set, to the command.

4.3.2. Deleting AVPs from a Command

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

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

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

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

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

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

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

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

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

4.3.3. Changing the Flags Setting of AVP in existing Commands

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

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

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

4.4. Reusing Existing AVPs

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

4.4.1. Setting of the AVP Flags

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

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

4.4.2. Reuse of AVP of Type Enumerated

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

5. Defining New Diameter Applications

5.1. Introduction

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

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

Additional considerations are described in the following sections.

5.2. Defining New Commands

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

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

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

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

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

5.3. Use of Application-Id in a Message

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

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

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

5.4. Application-Specific Session State Machines

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

5.5. Session-Id AVP and Session Management

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

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

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

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

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

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

5.6. Use of Enumerated Type AVPs

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

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

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

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

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

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

1001: 3GPP-GERAN

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

1002: 3GPP-UTRAN-FDD

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

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

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

5.7. Application-Specific Message Routing

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

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

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

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

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

5.8. Translation Agents

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

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

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

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

5.9. End-to-End Application Capabilities Exchange

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

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

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

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

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

5.10. Diameter Accounting Support

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

Split Accounting Model:

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

- * The application itself does not define its own accounting commands.
- * The overall system architecture permits the use of centralized accounting for one or more Diameter applications.

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

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

Coupled Accounting Model:

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

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

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

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

5.11. Diameter Security Mechanisms

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

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

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

6. Defining Generic Diameter Extensions

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

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

Backward Compatibility:

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

Forward Compatibility:

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

Trade-off in Signaling:

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

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

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

7. Guidelines for Registrations of Diameter Values

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

a. Defining new AVP values

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

b. Creating new AVPs

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

- * IETF AVP Codes namespace;
- * Vendor-specific AVP Codes namespace.

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

c. Creating new commands

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

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

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

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

d. Creating new applications

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

- * A range of values reserved for standard Application-Ids allocated after Expert Review of the specification defining the standard application;
- * A range for values for vendor specific applications, allocated by IANA on a First-Come/First-Serve basis.

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

NOTE:

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

8. IANA Considerations

This document does not require actions by IANA.

9. Security Considerations

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

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

10. Contributors

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

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

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

11. Acknowledgments

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

12. References

12.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.

12.2. Informative References

- [Q.3303.3] 3rd Generation Partnership Project, "ITU-T Recommendation Q.3303.3, "Resource control protocol no. 3 (rcp3): Protocol at the Rw interface between the Policy Decision Physical Entity (PD-PE) and the Policy Enforcement Physical Entity (PE-PE): Diameter"", 2008.
- [RFC2409] Harkins, D. and D. Carrel, "The Internet Key Exchange (IKE)", RFC 2409, November 1998.
- [RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.
- [RFC4005] Calhoun, P., Zorn, G., Spence, D., and D. Mitton, "Diameter Network Access Server Application", RFC 4005, August 2005.
- [RFC4072] Eronen, P., Hiller, T., and G. Zorn, "Diameter Extensible Authentication Protocol (EAP) Application", RFC 4072, August 2005.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.
- [RFC4740] Garcia-Martin, M., Belinchon, M., Pallares-Lopez, M., Canales-Valenzuela, C., and K. Tammi, "Diameter Session Initiation Protocol (SIP) Application", RFC 4740, November 2006.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.

- [RFC5447] Korhonen, J., Bournelle, J., Tschofenig, H., Perkins, C., and K. Chowdhury, "Diameter Mobile IPv6: Support for Network Access Server to Diameter Server Interaction", RFC 5447, February 2009.
- [RFC5777] Korhonen, J., Tschofenig, H., Arumaithurai, M., Jones, M., and A. Lior, "Traffic Classification and Quality of Service (QoS) Attributes for Diameter", RFC 5777, February 2010.
- [RFC5996] Kaufman, C., Hoffman, P., Nir, Y., and P. Eronen, "Internet Key Exchange Protocol Version 2 (IKEv2)", RFC 5996, September 2010.
- [RFC6083] Tuexen, M., Seggelmann, R., and E. Rescorla, "Datagram Transport Layer Security (DTLS) for Stream Control Transmission Protocol (SCTP)", RFC 6083, January 2011.
- [RFC6735] Carlberg, K. and T. Taylor, "Diameter Priority Attribute-Value Pairs", RFC 6735, October 2012.
- [RFC7075] Tsou, T., Hao, R., and T. Taylor, "Realm-Based Redirection In Diameter", RFC 7075, November 2013.
- [RFC7155] Zorn, G., "Diameter Network Access Server Application", RFC 7155, April 2014.
- [TS29.228] 3rd Generation Partnership Project, "3GPP TS 29.228; Technical Specification Group Core Network and Terminals; IP Multimedia (IM) Subsystem Cx and Dx Interfaces; Signalling flows and message contents", <<http://www.3gpp.org/ftp/Specs/html-info/29272.htm>>.
- [TS29.229] 3rd Generation Partnership Project, "3GPP TS 29.229; Technical Specification Group Core Network and Terminals; Cx and Dx interfaces based on the Diameter protocol; Protocol details", <<http://www.3gpp.org/ftp/Specs/html-info/29229.htm>>.
- [TS29.328] 3rd Generation Partnership Project, "3GPP TS 29.328; Technical Specification Group Core Network and Terminals; IP Multimedia (IM) Subsystem Sh interface; signalling flows and message content", <<http://www.3gpp.org/ftp/Specs/html-info/29328.htm>>.

[TS29.329]

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

Authors' Addresses

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

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

Victor Fajardo
Fluke Networks

Email: vf0213@gmail.com

Hannes Tschofenig
Hall in Tirol 6060
Austria

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

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

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

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

Abstract

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

Status of This Memo

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

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

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

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

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
2.1. Requirements Language	4
3. Assumptions	4
4. Protocol Overview	4
5. Bootstrapping the ER Server	6
5.1. Bootstrapping During the Initial EAP authentication	6
5.2. Bootstrapping During the First Re-authentication	8
6. Re-Authentication	10
7. Application Id	11
8. AVPs	12
8.1. ERP-RK-Request AVP	12
8.2. ERP-Realm AVP	12
8.3. Key AVP	12
8.3.1. Key-Type AVP	12
8.3.2. Keying-Material AVP	12
8.3.3. Key-Name AVP	13
8.3.4. Key-Lifetime AVP	13
9. Result-Code AVP Values	13
9.1. Permanent Failures	13
10. IANA Considerations	13
10.1. Diameter Application Identifier	13
10.2. New AVPs	13
10.3. New Permanent Failures Result-Code AVP Values	14
11. Security Considerations	14
12. Contributors	14
13. Acknowledgements	15
14. References	15
14.1. Normative References	15
14.2. Informative References	16

1. Introduction

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

Bootstrapping

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

Re-authentication

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

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

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

2. Terminology

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

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

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

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

located in the home or a foreign domain.

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

2.1. Requirements Language

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

3. Assumptions

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

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

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

4. Protocol Overview

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

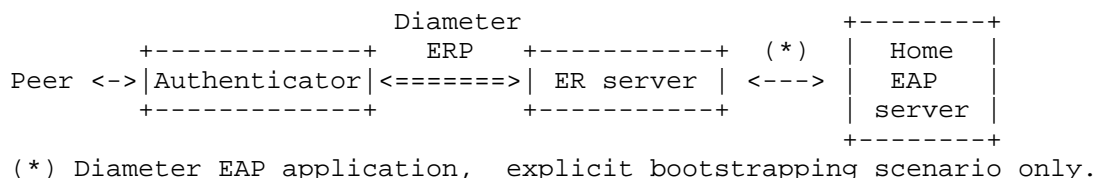


Figure 1: Diameter ERP Overview.

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

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

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

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

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

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

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

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

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

5. Bootstrapping the ER Server

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

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

5.1. Bootstrapping During the Initial EAP authentication

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

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

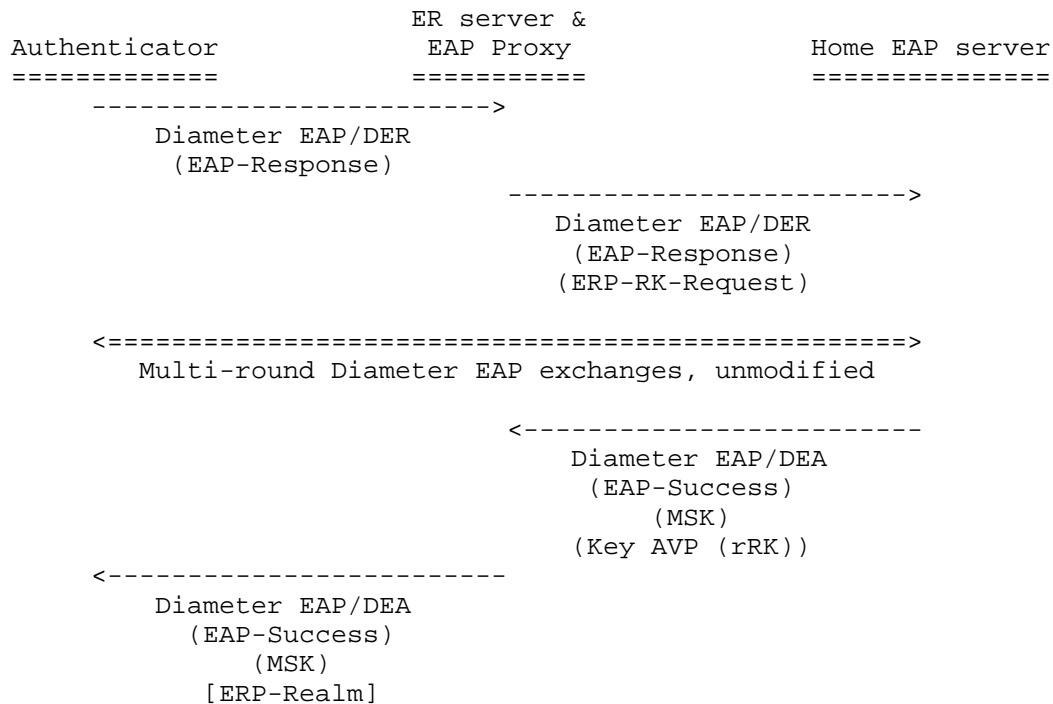


Figure 2: ERP Bootstrapping During Full EAP Authentication

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

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

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

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

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

5.2. Bootstrapping During the First Re-authentication

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

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

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

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

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

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

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

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

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

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

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

The figure below captures this proxy behavior:

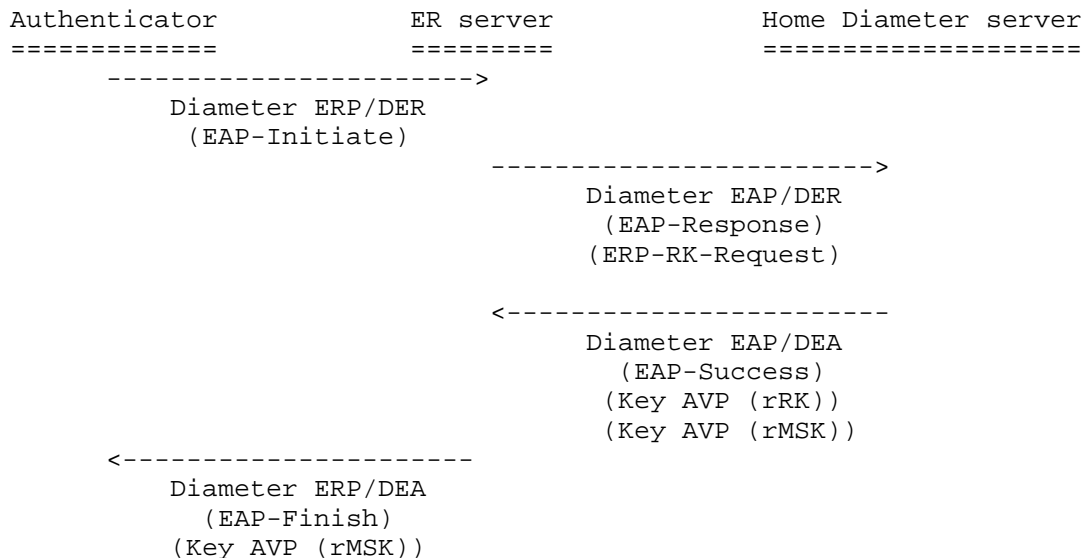


Figure 3: ERP Explicit Bootstrapping Message Flow

6. Re-Authentication

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

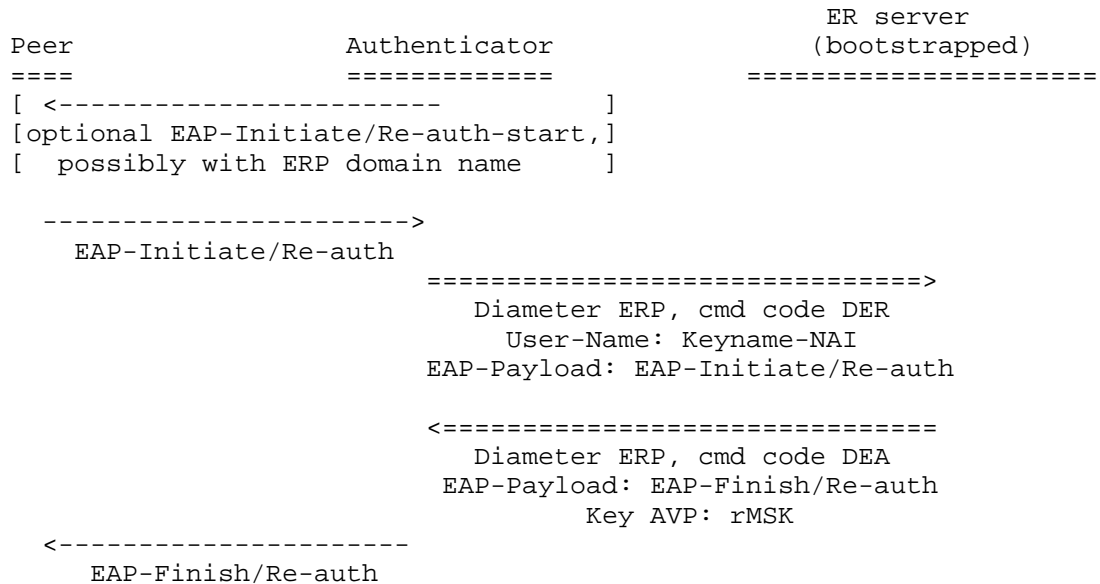


Figure 4: Diameter ERP Re-authentication Exchange

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

7. Application Id

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

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

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

8. AVPs

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

8.1. ERP-RK-Request AVP

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

This AVP has the M and V bits cleared.

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

Figure 5: ERP-RK-Request ABNF

8.2. ERP-Realm AVP

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

This AVP has the M and V bits cleared.

8.3. Key AVP

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

8.3.1. Key-Type AVP

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

8.3.2. Keying-Material AVP

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

8.3.3. Key-Name AVP

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

8.3.4. Key-Lifetime AVP

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

9. Result-Code AVP Values

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

9.1. Permanent Failures

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

DIAMETER_ERROR_EAP_CODE_UNKNOWN (TBD4)

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

10. IANA Considerations

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

10.1. Diameter Application Identifier

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

10.2. New AVPs

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

ERP-RK-Request (code: TBD2)

ERP-Realm (code: TBD3)

These AVPs are defined in Section 8.

10.3. New Permanent Failures Result-Code AVP Values

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

DIAMETER_ERROR_EAP_CODE_UNKNOWN (code: TBD4)

This result-code value is defined in Section 9.

11. Security Considerations

The security considerations from the following documents apply here:

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

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

12. Contributors

Hannes Tschofenig wrote the initial draft of this document.

Lakshminath Dondeti contributed to the early versions of the document.

13. Acknowledgements

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

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

Many thanks to these people!

14. References

14.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC3748] Aboba, B., Blunk, L., Vollbrecht, J., Carlson, J., and H. Levkowetz, "Extensible Authentication Protocol (EAP)", RFC 3748, June 2004.
- [RFC4072] Eronen, P., Hiller, T., and G. Zorn, "Diameter Extensible Authentication Protocol (EAP) Application", RFC 4072, August 2005.
- [RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, May 2008.
- [RFC5295] Salowey, J., Dondeti, L., Narayanan, V., and M. Nakhjiri, "Specification for the Derivation of Root Keys from an Extended Master Session Key (EMSK)", RFC 5295, August 2008.
- [RFC6696] Cao, Z., He, B., Shi, Y., Wu, Q., and G. Zorn, "EAP Extensions for the EAP Re-authentication Protocol (ERP)", RFC 6696, July 2012.
- [RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.
- [RFC6734] Zorn, G., Wu, Q., and V. Cakulev, "Diameter Attribute-Value Pairs for Cryptographic Key Transport", RFC 6734, October 2012.

14.2. Informative References

- [AAAPARAMS] Internet Assigned Numbers Authority, "Authentication, Authorization, and Accounting (AAA) Parameters", <http://www.iana.org/assignments/aaa-parameters/>.
- [RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.
- [RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.
- [RFC5246] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.
- [RFC6347] Rescorla, E. and N. Modadugu, "Datagram Transport Layer Security Version 1.2", RFC 6347, January 2012.

Authors' Addresses

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

EMail: julien.bournelle@orange-ftgroup.com

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

EMail: lionel.morand@orange.com

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

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

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

EMail: sunseawq@huawei.com

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

EMail: glenzorn@gmail.com

Diameter Maintenance and Extensions (DIME)
Internet-Draft
Intended status: Standards Track
Expires: September 10, 2020

M. Jones
M. Liebsch
L. Morand
March 9, 2020

Diameter Group Signaling
draft-ietf-dime-group-signaling-13.txt

Abstract

In large network deployments, a single Diameter node can support over a million concurrent Diameter sessions. Recent use cases have revealed the need for Diameter nodes to apply the same operation to a large group of Diameter sessions concurrently. The Diameter base protocol commands operate on a single session so these use cases could result in many thousands of command exchanges to enforce the same operation on each session in the group. In order to reduce signaling, it would be desirable to enable bulk operations on all (or part of) the sessions managed by a Diameter node using a single or a few command exchanges. This document specifies the Diameter protocol extensions to achieve this signaling optimization.

Status of This Memo

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

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

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

This Internet-Draft will expire on September 10, 2020.

Copyright Notice

Copyright (c) 2020 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<https://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. Protocol Overview	4
3.1. Building and Modifying Session Groups	4
3.2. Issuing Group Commands	4
3.3. Permission Considerations	5
4. Protocol Description	6
4.1. Session Grouping Capability Discovery	6
4.1.1. Implicit Capability Discovery	6
4.1.2. Explicit Capability Discovery	7
4.2. Session Grouping	7
4.2.1. Group assignment at session initiation	8
4.2.2. Removing a session from a session group	10
4.2.3. Mid-session group assignment modifications	12
4.3. Deleting a Session Group	13
4.4. Performing Group Operations	13
4.4.1. Sending Group Commands	13
4.4.2. Receiving Group Commands	14
4.4.3. Error Handling for Group Commands	14
4.4.4. Single-Session Fallback	15
5. Operation with Proxy Agents	15
6. Commands Formatting	16
6.1. Formatting Example: Group Re-Auth-Request	16
7. Attribute-Value-Pairs (AVP)	17
7.1. Session-Group-Info AVP	17
7.2. Session-Group-Control-Vector AVP	18
7.3. Session-Group-Id AVP	18
7.4. Group-Response-Action AVP	19
7.5. Session-Group-Capability-Vector AVP	19
8. Result-Code AVP Values	19
9. IANA Considerations	19
9.1. AVP Codes	20
9.2. New Registries	20
10. Security Considerations	20
11. Acknowledgments	21
12. Normative References	21

Appendix A. Session Management -- Exemplary Session State	
Machine	22
A.1. Use of groups for the Authorization Session State Machine	22
Authors' Addresses	26

1. Introduction

In large network deployments, a single Diameter node can support over a million concurrent Diameter sessions. Recent use cases have revealed the need for Diameter nodes to apply the same operation to a large group of Diameter sessions concurrently. For example, a policy decision point may need to modify the authorized quality of service for all active users having the same type of subscription. The Diameter base protocol commands operate on a single session so these use cases could result in many thousands of command exchanges to enforce the same operation on each session in the group. In order to reduce signaling, it would be desirable to enable bulk operations on all (or part of) the sessions managed by a Diameter node using a single or a few command exchanges.

This document describes mechanisms for grouping Diameter sessions and applying Diameter commands, such as performing re-authentication, re-authorization, termination and abortion of sessions to a group of sessions. This document does not define a new Diameter application. Instead it defines mechanisms, commands and AVPs that may be used by any Diameter application that requires management of groups of sessions.

These mechanisms take the following design goals and features into account:

- o Minimal impact to existing applications
- o Extension of existing commands' Command Code Format (CCF) with optional AVPs to enable grouping and group operations
- o Fallback to single session operation
- o Implicit discovery of capability to support grouping and group operations in case no external mechanism is available to discover a Diameter peer's capability to support session grouping and session group operations

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP

14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This document uses terminology defined in [RFC6733].

3. Protocol Overview

3.1. Building and Modifying Session Groups

Client and Server can assign a new Diameter session to a group, e.g., in case the subscription profile of the associated user has similar characteristics as the profile of other users whose Diameter session has been assigned to one or multiple groups. A single command can be issued and applied to all sessions associated with such group(s), e.g., to adjust common profile or policy settings.

The assignment of a Diameter session to a group can be changed during an ongoing session (mid-session). For example, if a user's subscription profile changes mid-session, a Diameter server may remove a session from an existing group and assign this session to a different group that is more appropriate for the new subscription profile.

In the case of mobile users, the user's session may get transferred mid-session to a new Diameter client during handover and assigned to a different group, which is maintained at the new Diameter client.

A session group, which has sessions assigned, can be deleted, e.g., due to a change in multiple users' subscription profile so that the group's assigned sessions do not share certain characteristics anymore. Deletion of such group requires subsequent individual treatment of each of the assigned sessions. A node may decide to assign some of these sessions to any other existing or new group.

3.2. Issuing Group Commands

Changes in the network condition may result in the Diameter server's decision to close all sessions in a given group. As example, the server issues a single Session Termination Request (STR) command, including the identifier of the group of sessions which are to be terminated. The Diameter client treats the STR as group command and initiates the termination of all sessions associated with the identified group. Subsequently, the client confirms the successful termination of these sessions to the server by sending a single Session Termination Answer (STA) command, which includes the identifier of the group.

3.3. Permission Considerations

Permission considerations in the context of this draft apply to the permission of Diameter nodes to build new session groups, to assign/remove a session to/from a session group and to delete an existing session group.

This specification follows the most flexible model where both, a Diameter client and a Diameter server can create a new group and assign a new identifier to that session group. When a client or a server decides to create a new session group, e.g., to group all sessions which share certain characteristics, this node builds a session group identifier according to the rules described in Section 7.3 and becomes the owner of the group. This specification does not restrict the permission to add or remove a session to/from a session group to the group owner. Either the client and the server can assign a session to a group. However, a session can be removed from a session group and/or moved to another session group only by the node that has assigned this session to the session group. A session group is deleted and its identifier released after the last session has been removed from this session group. The owner of a session group can delete a session group and its group identifier mid-session, resulting in individual treatment of the sessions which have been previously assigned to the deleted group. A session group must only be deleted by the Diameter node that created it.

Diameter applications with implicit support for session groups MAY define a more constrained permission model. For example, a more constrained model could require that a client must not remove a session from a group which is owned by the server. Details about enforcing a more constraint permission model are out of scope of this specification.

The following table depicts the permission considerations as per the present specification:

Operation	Server	Client
Create a new Session Group (Diameter node becomes the group owner)	X	X
Assign a Session to an owned Session Group	X	X
Assign a Session to a non-owned Session Group	X	X
Remove a Session from an owned Session Group	X	X
Remove a Session from a non-owned Session Group	X	X
Remove a Session from a Session Group where the Diameter node created the assignment	X	X
Remove a Session from a Session Group where the Diameter node did not create the assignment		
Overrule a different Diameter node's group assignment *)		
Delete a Session Group which is owned by the Diameter node	X	X
Delete a Session Group which is not owned by the Diameter node		

Default Permission as per this Specification

4. Protocol Description

4.1. Session Grouping Capability Discovery

Diameter nodes SHOULD NOT perform group operations with peer nodes unless the node has advertised support for session grouping and group operations.

4.1.1. Implicit Capability Discovery

Newly defined Diameter applications may natively support Diameter session grouping and group operations. Such applications provide intrinsic discovery for the support of session grouping capability using the assigned Application Id advertised during the capability

exchange phase two Diameter peers establish a transport connection (see Section 5.3 of [RFC6733]).

System- and deployment-specific means, as well as out-of-band mechanisms for capability discovery can be used to announce nodes' support for session grouping and session group operations. In such case, the optional Session-Group-Capability-Vector AVP, as described in Section 4.1.2 can be omitted in Diameter messages being exchanged between nodes.

4.1.2. Explicit Capability Discovery

If no other mechanism for capability discovery is deployed to enable Diameter nodes to learn about nodes' capability to support session grouping and group commands for a given application, a Diameter node SHOULD append the Session-Group-Capability-Vector AVP to any Diameter application messages exchanged with the other Diameter nodes to announce its capability to support session grouping and session group operations for the advertised application. Implementations following the specification as per this document MUST set the BASE_SESSION_GROUP_CAPABILITY flag of the Session-Group-Capability-Vector AVP.

When a Diameter node receives at least one Session-Group-Capability-Vector AVP from a node with the BASE_SESSION_GROUP_CAPABILITY flag set, the receiving Diameter node discovers the supported session grouping capability of the sending Diameter node for the advertised application and MUST cache this information for the lifetime of the routing table entry associated with the peer identity/Application Id pair (see Section 2.7 of [RFC6733]).

4.2. Session Grouping

This specification does not limit the number of session groups to which a single session is assigned. It is left to the implementation of an application to determine such limitations. If an application facilitates a session to belong to multiple session groups, the application MUST maintain consistency of associated application session states for these multiple session groups.

Either Diameter node (client or server) can initiate the assignment of a session to a single or multiple session groups. Modification of a group by removing or adding a single or multiple user sessions can be initiated and performed mid-session by either Diameter node responsible for the session assignment to this group. Diameter AAA applications typically assign client and server roles to the Diameter nodes, which are referred to as relevant Diameter nodes to utilize session grouping and issue group commands. Section 5 describes

particularities about session grouping and performing group commands when relay agents or proxies are deployed.

Diameter nodes, which are group-aware, MUST store and maintain an entry about the group assignment together with a session's state. A list of all known session groups is locally maintained on each node, each group pointing to individual sessions being assigned to the group. A Diameter node MUST also keep a record about sessions, which have been assigned to a session group by itself.

4.2.1. Group assignment at session initiation

To assign a session to a group at session initiation, a Diameter client sends a service specific request, e.g., NASREQ AA-Request [RFC7155], containing one or more session group identifiers. Each of these groups MUST be identified by a unique Session-Group-Id contained in a separate Session-Group-Info AVP as specified in Section 7.

The client may choose one or multiple session groups from a list of existing session groups. Alternatively, the client may decide to create a new group to which the session is assigned and identify itself in the <DiameterIdentity> portion of the Session-Group-Id AVP as per Section 7.3. For all assignments of a session to an active session group made by the client or the server, the SESSION_GROUP_STATUS_IND flag in the Session-Group-Info AVP, which identifies the session group, MUST be set. A set SESSION_GROUP_STATUS_IND flag indicates that the identified session group has just been created or is still active.

The client MUST set the SESSION_GROUP_ALLOCATION_ACTION flag of the Session-Group-Control-Vector AVP in each appended Session-Group-Info AVP to indicate that the session contained in the request should be assigned to the identified session group.

The client may also indicate in the request that the server is responsible for the assignment of the session in one or multiple sessions owned by the server. In such a case, the client MUST include the Session-Group-Info AVP in the request including the Session-Group-Control-Vector AVP with the SESSION_GROUP_ALLOCATION_ACTION flag set but no Session-Group-Id AVP.

If the Diameter server receives a command request from a Diameter client and the command includes at least one Session-Group-Info AVP having the SESSION_GROUP_ALLOCATION_ACTION flag in the Session-Group-Control-Vector AVP set, the server can accept or reject the request for group assignment. Reasons for rejection may be e.g., lack of

resources for managing additional groups. When rejected, the session MUST NOT be assigned to any session group.

If the Diameter server accepts the client's request for a group assignment, the server MUST assign the new session to each of the one or multiple identified session groups when present in the Session-Group-Info AVP. If one or multiple identified session groups are not already stored by the server, the server MUST store the newly identified group(s) to its local list of known session groups. When sending the response to the client, e.g., a service-specific auth response as per NASREQ AA-Answer [RFC7155], the server MUST include all Session-Group-Info AVPs as received in the client's request.

In addition to the one or multiple session groups identified in the client's request, the server may decide to assign the new session to one or multiple additional groups. In such a case, the server MUST add to the response the additional Session-Group-Info AVPs, each identifying a session group to which the new session is assigned by the server. Each of the Session-Group-Info AVP added by the server MUST have the SESSION_GROUP_ALLOCATION_ACTION flag set in the Session-Group-Control-Vector AVP set.

If the Diameter server rejects the client's request for a group assignment, the server sends the response to the client, e.g., a service-specific auth response as per NASREQ AA-Answer [RFC7155], and MUST include all Session-Group-Info AVPs as received in the client's request (if any) while clearing the SESSION_GROUP_ALLOCATION_ACTION flag of the Session-Group-Control-Vector AVP. The server MAY accept the client's request for the identified session but refuse the session's assignment to any session group. The server sends the response to the client indicating success in the result code. In such case the session is treated as single session without assignment to any session group by the Diameter nodes.

If the assignment of the session to one or some of the multiple identified session groups fails, the session group assignment is treated as failure. In such case the session is treated as single session without assignment to any session group by the Diameter nodes. The server sends the response to the client and MAY include those Session-Group-Info AVPs for which the group assignment failed. The SESSION_GROUP_ALLOCATION_ACTION flag of included Session-Group-Info AVPs MUST be cleared.

If the Diameter server receives a command request from a Diameter client and the command includes a Session-Group-Info AVP which does not include a Session-Group-Id AVP, the server MAY decide to assign the session to one or multiple session groups. For each session group, to which the server assigns the new session, the server

includes a Session-Group-Info AVP with the Session-Group-Id AVP identifying a session group in the response sent to the client. Each of the Session-Group-Info AVPs included by the server MUST have the SESSION_GROUP_ALLOCATION_ACTION flag of the Session-Group-Control-Vector AVP set.

If the Diameter server receives a command request from a Diameter client and the command does not contain any Session-Group-Info AVP, the server MUST NOT assign the new session to any session group but treat the request as for a single session. The server MUST NOT return any Session-Group-Info AVP in the command response.

If the Diameter client receives a response to its previously issued request from the server and the response includes at least one Session-Group-Info AVP having the SESSION_GROUP_ALLOCATION_ACTION flag of the associated Session-Group-Control-Vector AVP set, the client MUST add the new session to all session groups as identified in the one or multiple Session-Group-Info AVPs. If the Diameter client fails to add the session to one or more session groups as identified in the one or multiple Session-Group-info AVPs, the client MUST terminate the session. The client MAY send a subsequent request for session initiation to the server without requesting the assignment of the session to a session group

If the Diameter client receives a response to its previously issued request from the server and the one or more Session-Group-Info AVPs have the SESSION_GROUP_ALLOCATION_ACTION flag of the associated Session-Group-Control-Vector AVP cleared, the client MUST terminate the assignment of the session to the one or multiple groups. If the response from the server indicates success in the result code but solely the assignment of the session to a session group has been rejected by the server, the client treats the session as single session without group assignment.

A Diameter client, which sent a request for session initiation to a Diameter server and appended a single or multiple Session-Group-Id AVPs but cannot find any Session-Group-Info AVP in the associated response from the Diameter server proceeds as if the request was processed for a single session. The Diameter client MUST NOT retry to request group assignment for this session, but MAY try to request group assignment for other new sessions.

4.2.2. Removing a session from a session group

When a Diameter client decides to remove a session from a particular session group, the client sends a service-specific re-authorization request to the server and adds one Session-Group-Info AVP to the request for each session group, from which the client wants to remove

the session. The session, which is to be removed from a group, is identified in the Session-Id AVP of the command request. The SESSION_GROUP_ALLOCATION_ACTION flag of the Session-Group-Control-Vector AVP in each Session-Group-Info AVP MUST be cleared to indicate removal of the session from the session group identified in the associated Session-Group-id AVP.

When a Diameter client decides to remove a session from all session groups, to which the session has been previously assigned, the client sends a service-specific re-authorization request to the server and adds a single Session-Group-Info AVP to the request which has the SESSION_GROUP_ALLOCATION_ACTION flag cleared and the Session-Group-Id AVP omitted. The session, which is to be removed from all groups, to which the session has been previously assigned, is identified in the Session-Id AVP of the command request.

If the Diameter server receives a request from the client which has at least one Session-Group-Info AVP appended with the SESSION_GROUP_ALLOCATION_ACTION flag cleared, the server MUST remove the session from the session group identified in the associated Session-Group-Id AVP. If the request includes at least one Session-Group-info AVP with the SESSION_GROUP_ALLOCATION_ACTION flag cleared and no Session-Id AVP present, the server MUST remove the session from all session groups to which the session has been previously assigned. The server MUST include in its response to the requesting client all Session-Group-Id AVPs as received in the request.

When the Diameter server decides to remove a session from one or multiple particular session groups or from all session groups to which the session has been assigned beforehand, the server sends a Re-Authorization Request (RAR) or a service-specific server-initiated request to the client, indicating the session in the Session-Id AVP of the request. The client sends a Re-Authorization Answer (RAA) or a service-specific answer to respond to the server's request. The client subsequently sends service-specific re-authorization request containing one or multiple Session-Group-Info AVPs, each indicating a session group, to which the session had been previously assigned. To indicate removal of the indicated session from one or multiple session groups, the server sends a service-specific auth response to the client, containing a list of Session-Group-Info AVPs with the SESSION_GROUP_ALLOCATION_ACTION flag cleared and the Session-Group-Id AVP identifying the session group, from which the session should be removed. The server MAY include to the service-specific auth response a list of Session-Group-Info AVPs with the SESSION_GROUP_ALLOCATION_ACTION flag set and the Session-Group-Id AVP identifying session groups to which the session remains subscribed. If the server decides to remove the identified session from all session groups, to which the session has been previously assigned,

the server includes in the service-specific auth response at least one Session-Group-Info AVP with the SESSION_GROUP_ALLOCATION_ACTION flag cleared and Session-Group-Id AVP absent.

4.2.3. Mid-session group assignment modifications

Either Diameter node (client or server) can modify the group membership of an active Diameter session according to the specified permission considerations.

To update an assigned group mid-session, a Diameter client sends a service-specific re-authorization request to the server, containing one or multiple Session-Group-Info AVPs with the SESSION_GROUP_ALLOCATION_ACTION flag set and the Session-Group-Id AVP present, identifying the session group to which the session should be assigned. With the same message, the client may send one or multiple Session-Group-Info AVP with the SESSION_GROUP_ALLOCATION_ACTION flag cleared and the Session-Group-Id AVP identifying the session group from which the identified session is to be removed. To remove the session from all previously assigned session groups, the client includes at least one Session-Group-Info AVP with the SESSION_GROUP_ALLOCATION_ACTION flag cleared and no Session-Group-Id AVP present. When the server received the service-specific re-authorization request, it MUST update its locally maintained view of the session groups for the identified session according to the appended Session-Group-Info AVPs. The server sends a service-specific auth response to the client containing one or multiple Session-Group-Info AVPs with the SESSION_GROUP_ALLOCATION_ACTION flag set and the Session-Group-Id AVP identifying the new session group to which the identified session has been assigned.

When a Diameter server enforces an update to the assigned groups mid-session, it sends a Re-Authorization Request (RAR) message or a service-specific request to the client identifying the session, for which the session group lists are to be updated. The client responds with a Re-Authorization Answer (RAA) message or a service-specific answer. The client subsequently sends a service-specific re-authorization request containing one or multiple Session-Group-Info AVPs with the SESSION_GROUP_ALLOCATION_ACTION flag set and the Session-Group-Id AVP identifying the session group to which the session had been previously assigned. The server responds with a service-specific auth response and includes one or multiple Session-Group-Info AVP with the SESSION_GROUP_ALLOCATION_ACTION flag set and the Session-Group-Id AVP identifying the session group, to which the identified session is to be assigned. With the same response message, the server may send one or multiple Session-Group-Info AVPs with the SESSION_GROUP_ALLOCATION_ACTION flag cleared and the Session-Group-Id AVP identifying the session groups from which the

identified session is to be removed. When server wants to remove the session from all previously assigned session groups, it sends at least one Session-Group-Info AVP with the response having the SESSION_GROUP_ALLOCATION_ACTION flag cleared and no Session-Group-Id AVP present.

4.3. Deleting a Session Group

To delete a session group and release the associated Session-Group-Id value, the owner of a session group appends a single Session-Group-Info AVP having the SESSION_GROUP_STATUS_IND flag cleared and the Session-Group-Id AVP identifying the session group, which is to be deleted. The SESSION_GROUP_ALLOCATION_ACTION flag of the associated Session-Group-Control-Vector AVP MUST be cleared.

4.4. Performing Group Operations

4.4.1. Sending Group Commands

Either Diameter node (client or server) can request the recipient of a request to process an associated command for all sessions assigned to one or multiple groups by identifying these groups in the request. The sender of the request appends for each group, to which the command applies, a Session-Group-Info AVP including the Session-Group-Id AVP to identify the associated session group. Both, the SESSION_GROUP_ALLOCATION_ACTION flag as well as the SESSION_GROUP_STATUS_IND flag MUST be set.

If the Command Code Format (CCF) of the request mandates a Session-Id AVP, the Session-Id AVP MUST identify one of the single sessions which is assigned to at least one of the groups being identified in the appended Session-Group-Id AVPs.

The sender of the request MUST indicate to the receiver how multiple resulting transactions associated with a group command are to be treated by appending a single instance of a Group-Response-Action AVP. For example, when a server sends a Re-Authorization Request (RAR) or a service-specific server-initiated request to the client, it indicates to the client to follow the request according to one of three possible procedures. When the server sets the Group-Response-Action AVP to ALL_GROUPS (1), the client sends a single RAR message for all identified groups. When the server sets the Group-Response-Action AVP to PER_GROUP (2), the client sends a single RAR message for each identified group individually. When the server sets the Group-Response-Action AVP to PER_SESSION (3), the client follows-up with a single RAR message per impacted session. If a session is included in more than one of the identified session groups, the client sends only one RAR message for that session.

If the sender sends a request including the Group-Response-Action AVP set to ALL_GROUPS (1) or PER_GROUP (2), it has to expect some delay before receiving the corresponding answer(s) as the answer(s) will only be sent back when the request is processed for all the sessions or all the session of a session group. If the process of the request is delay-sensitive, the sender SHOULD NOT set the Group-Response-Action AVP to ALL_GROUPS (1) or PER_GROUP (2). If the answer can be sent before the complete process of the request for all the sessions or if the request timeout timer is high enough, the sender MAY set the Group-Response-Action AVP to ALL_GROUPS (1) or PER_GROUP (2).

If the sender wants the receiver of the request to process the associated command solely for a single session, the sender does not append any group identifier, but identifies the relevant session in the Session-Id AVP.

4.4.2. Receiving Group Commands

A Diameter node receiving a request to process a command for a group of sessions, identifies the relevant groups according to the appended Session-Group-Id AVP in the Session-Group-Info AVP and processes the group command according to the appended Group-Response-Action AVP. If the received request identifies multiple groups in multiple appended Session-Group-Id AVPs, the receiver SHOULD process the associated command for each of these groups. If a session has been assigned to more than one of the identified groups, the receiver MUST process the associated command only once per session.

4.4.3. Error Handling for Group Commands

When a Diameter node receives a request to process a command for one or more session groups and the result of processing the command is an error that applies to all sessions in the identified groups, an associated protocol error MUST be returned to the source of the request. In such case, the sender of the request MUST fall back to single-session processing and the session groups, which have been identified in the group command, MUST be deleted according to the procedure described in Section 4.3.

When a Diameter node receives a request to process a command for one or more session groups and the result of processing the command succeeds for some sessions identified in one or multiple session groups, but fails for one or more sessions, the Result-Code AVP in the response message SHOULD indicate DIAMETER_LIMITED_SUCCESS as per Section 7.1.2 of [RFC6733].

In the case of limited success, the sessions, for which the processing of the group command failed, MUST be identified using a

Failed-AVP AVP as per Section 7.5 of [RFC6733]. The sender of the request MUST fall back to single-session operation for each of the identified sessions, for which the group command failed. In addition, each of these sessions MUST be removed from all session groups to which the group command applied. To remove sessions from a session group, the Diameter client performs the procedure described in Section 4.2.2.

4.4.4. Single-Session Fallback

Either Diameter node can fall back to single session operation by ignoring and omitting the optional group session-specific AVPs. Fallback to single-session operation is performed by processing the Diameter command solely for the session identified in the mandatory Session-Id AVP. In such case, the response to the group command MUST NOT identify any group but identify solely the single session for which the command has been processed.

5. Operation with Proxy Agents

In the case of a present stateful Proxy Agent between a Diameter client and a Diameter server, the Proxy Agent MUST perform the same mechanisms per this specification to advertise session grouping and group operations capability towards the client and the server respectively. The Proxy MUST update and maintain consistency of its local session states as per the result of the group commands which are operated between a Diameter client and a server. In such case, the Proxy Agent MUST act as a Diameter server in front of the Diameter client and MUST act as a Diameter client in front of the Diameter server. Therefore, the client and server behavior described in Section 4 applies respectively to the stateful Proxy Agent.

If a stateful Proxy Agent manipulates session groups, it MUST maintain consistency of session groups between a client and a server. This applies to a deployment where the Proxy Agent utilizes session grouping and performs group operations with, for example, a Diameter server, whereas the Diameter client is not aware of session groups. In such case the Proxy Agent must reflect the states associated with the session groups as individual session operations towards the client and ensure the client has a consistent view of each session. The same applies to a deployment where all nodes, the Diameter client and server, as well as the Proxy Agent are group-aware but the Proxy Agent manipulates groups, e.g., to adopt different administrative policies that apply to the client's domain and the server's domain.

Stateless Proxy Agents do not maintain any session state (only transaction state are maintained). Consequently, the notion of session group is transparent for any stateless Proxy Agent present

between a Diameter client and a Diameter server handling session groups. Session group related AVPs being defined as optional AVP are ignored by stateless Proxy Agents and should not be removed from the Diameter commands. If they are removed by the Proxy Agent for any reason, the Diameter client and Diameter server will discover the absence the related session group AVPs and will fall back to single-session processing, as described in Section 4.

6. Commands Formatting

This document does not specify new Diameter commands to enable group operations, but relies on command extensibility capability provided by the Diameter Base protocol. This section provides the guidelines to extend the CCF of existing Diameter commands with optional AVPs to enable the recipient of the command applying the command to all sessions associated with the identified group(s).

6.1. Formatting Example: Group Re-Auth-Request

A request for re-authentication of one or more groups of users is issued by appending one or multiple Session-Group-Id AVP(s), as well as a single instance of a Group-Response-Action AVP to the Re-Auth-Request (RAR). The one or multiple Session-Group-Id AVP(s) identify the associated group(s) for which the group re-authentication has been requested. The Group-Response-Action AVP identifies the expected means to perform and respond to the group command. The recipient of the group command initiates re-authentication for all users associated with the identified group(s). Furthermore, the sender of the group re-authentication request appends a Group-Response-Action AVP to provide more information to the receiver of the command about how to accomplish the group operation.

The value of the mandatory Session-Id AVP MUST identify a session associated with a single user, which is assigned to at least one of the groups being identified in the appended Session-Group-Id AVPs.

```

<RAR> ::= < Diameter Header: 258, REQ, PXY >
        < Session-Id >
        { Origin-Host }
        { Origin-Realm }
        { Destination-Realm }
        { Destination-Host }
        { Auth-Application-Id }
        { Re-Auth-Request-Type }
        [ User-Name ]
        [ Origin-State-Id ]
        * [ Proxy-Info ]
        * [ Route-Record ]
        [ Session-Group-Capability-Vector ]
        * [ Session-Group-Info ]
        [ Group-Response-Action ]
        * [ AVP ]

```

7. Attribute-Value-Pairs (AVP)

Attribute Name	AVP Code Value Type	AVP Flag rules			
		MUST	MAY	SHOULD NOT	MUST NOT
Session-Group-Info	TBD1 Grouped		P		V
Session-Group-Control-Vector	TBD2 Unsigned32		P		V
Session-Group-Id	TBD3 OctetString		P		V
Group-Response-Action	TBD4 Unsigned32		P		V
Session-Group-Capability-Vector	TBD5 Unsigned32		P		V

AVPs for the Diameter Group Signaling

7.1. Session-Group-Info AVP

The Session-Group-Info AVP (AVP Code TBD1) is of type Grouped. It contains the identifier of the session group as well as an indication of the node responsible for session group identifier assignment.

```

Session-Group-Info ::= < AVP Header: TBD1 >
                        < Session-Group-Control-Vector >
                        [ Session-Group-Id ]
                        * [ AVP ]

```

7.2. Session-Group-Control-Vector AVP

The Session-Group-Control-Vector AVP (AVP Code TBD2) is of type Unsigned32 and contains a 32-bit flags field to control the group assignment at session-group aware nodes.

The following control flags are defined in this document:

SESSION_GROUP_ALLOCATION_ACTION (0x00000001)

This flag indicates the action to be performed for the identified session. When this flag is set, it indicates that the identified Diameter session is to be assigned to the session group as identified by the Session-Group-Id AVP or the session's assignment to the session group identified in the Session-Group-Id AVP is still valid. When the flag is cleared, the identified Diameter session is to be removed from at least one session group. When the flag is cleared and the Session-Group-Info AVP identifies a particular session group in the associated Session-Group-Id AVP, the session is to be removed solely from the identified session group. When the flag is cleared and the Session-Group-Info AVP does not identify a particular session group (Session-Group-Id AVP is absent), the identified Diameter session is to be removed from all session groups, to which it has been previously assigned.

SESSION_GROUP_STATUS_IND (0x00000010)

This flag indicates the status of the session group identified in the associated Session-Group-Id AVP. The flag is set when the identified session group has just been created or is still active. If the flag is cleared, the identified session group is deleted and the associated Session-Group-Id is released. If the Session-Group-Info AVP does not include a Session-Group-Id AVP, this flag is meaningless and MUST be ignored by the receiver.

7.3. Session-Group-Id AVP

The Session-Group-Id AVP (AVP Code TBD3) is of type UTF8String and identifies a group of Diameter sessions.

The Session-Group-Id MUST be globally unique. The default format of the Session-Group-id MUST comply to the format recommended for a Session-Id, as defined in the section 8.8 of the [RFC6733]. The <DiameterIdentity> portion of the Session-Group-Id MUST identify the Diameter node, which owns the session group.

7.4. Group-Response-Action AVP

The Group-Response-Action AVP (AVP Code TBD4) is of type Unsigned32 and contains a 32-bit address space representing values indicating how the node SHOULD issue follow up exchanges in response to a command which impacts multiple sessions. The following values are defined by this document:

ALL_GROUPS (1)

Follow up message exchanges associated with a group command should be performed with a single message exchange for all impacted groups.

PER_GROUP (2)

Follow up message exchanges associated with a group command should be performed with a separate message exchange for each impacted group.

PER_SESSION (3)

Follow up message exchanges associated with a group command should be performed with a separate message exchange for each impacted session.

7.5. Session-Group-Capability-Vector AVP

The Session-Group-Capability-Vector AVP (AVP Code TBD5) is of type Unsigned32 and contains a 32-bit flags field to indicate capabilities in the context of session-group assignment and group operations.

The following capabilities are defined in this document:

BASE_SESSION_GROUP_CAPABILITY (0x00000001)

This flag indicates the capability to support session grouping and session group operations according to this specification.

8. Result-Code AVP Values

This document does not define new Result-Code [RFC6733] values for existing applications, which are extended to support group commands. Specification documents of new applications, which will have intrinsic support for group commands, may specify new Result-Codes.

9. IANA Considerations

This section contains the namespaces that have either been created in this specification or had their values assigned to existing namespaces managed by IANA.

9.1. AVP Codes

This specification requires IANA to register the following new AVPs from the AVP Code namespace defined in [RFC6733].

- o Session-Group-Info
- o Session-Group-Control-Vector
- o Session-Group-Id
- o Group-Response-Action
- o Session-Group-Capability-Vector

The AVPs are defined in Section 7.

9.2. New Registries

This specification requires IANA to create two registries:

- o Session-Group-Control-Vector AVP registry for control bits with two initial assignments, which are described in Section 7.2. The future registration assignment policy is proposed to be Specification Required.
- o Session-Group-Capability-Vector AVP with one initial assignment, which is described in Section 7.5. The future registration assignment policy is proposed to be Standards Action.

The AVP names can be used as registry names.

10. Security Considerations

The security considerations of the Diameter protocol itself are discussed in [RFC6733]. Use of the AVPs defined in this document MUST take into consideration the security issues and requirements of the Diameter base protocol. In particular, the Session-Group-Info AVP (including the Session-group-Control-Vector and the Session-Group-Id AVPs) should be considered as a security-sensitive AVPs in the same manner than the Session-Id AVP in the Diameter base protocol [RFC6733].

The management of session groups relies upon the existing trust relationship between the Diameter client and the Diameter server managing the groups of sessions. This document defines a mechanism that allows a client or a server to act on multiple sessions at the same time using only one command. if the Diameter client or server is

compromised, an attacker could launch DoS attacks by terminating a large number of sessions with a limited set of commands using the session group management concept.

According to the Diameter base protocol [RFC6733], transport connections between Diameter peers are protected by TLS/TCP, DTLS/SCTP or alternative security mechanisms that are independent of Diameter, such as IPsec. However, the lack of end-to-end security features makes it difficult to establish trust in the session group related information received from non-adjacent nodes. Any Diameter agent in the message path can potentially modify the content of the message and therefore the information sent by the Diameter client or the server. There is ongoing work on the specification of end-to-end security features for Diameter. Such features would enable the establishment of trust relationship between non-adjacent nodes and the security required for session group management would normally rely on this end-to-end security. However, there is no assumption in this document that such end-to-end security mechanism will be available. It is only assumed that the solution defined on this document relies on the security framework provided by the Diameter based protocol.

In some cases, a Diameter Proxy agent can act on behalf of a client or server. In such a case, the security requirements that normally apply to a client (or a server) apply equally to the Proxy agent.

11. Acknowledgments

The authors of this document want to thank Ben Campbell and Eric McMurry for their valuable comments to early versions of this draft. Furthermore, authors thank Steve Donovan and Mark Bales for the thorough review and comments on advanced versions of the WG document, which helped a lot to improve this specification.

12. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <<https://www.rfc-editor.org/info/rfc2119>>.
- [RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <<https://www.rfc-editor.org/info/rfc8174>>.

[RFC6733] Fajardo, V., Ed., Arkko, J., Loughney, J., and G. Zorn, Ed., "Diameter Base Protocol", RFC 6733, DOI 10.17487/RFC6733, October 2012, <<https://www.rfc-editor.org/info/rfc6733>>.

[RFC7155] Zorn, G., Ed., "Diameter Network Access Server Application", RFC 7155, DOI 10.17487/RFC7155, April 2014, <<https://www.rfc-editor.org/info/rfc7155>>.

Appendix A. Session Management -- Exemplary Session State Machine

A.1. Use of groups for the Authorization Session State Machine

Section 8.1 in [RFC6733] defines a set of finite state machines, representing the life cycle of Diameter sessions, and which must be observed by all Diameter implementations that make use of the authentication and/or authorization portion of a Diameter application. This section defines, as example, additional state transitions related to the processing of the group commands which may impact multiple sessions.

The group membership is session state and therefore only those state machines from [RFC6733] in which the server is maintaining session state are relevant in this document. As in [RFC6733], the term Service-Specific below refers to a message defined in a Diameter application (e.g., Mobile IPv4, NASREQ).

The following state machine is observed by a client when state is maintained on the server. State transitions which are unmodified from [RFC6733] are not repeated here.

The Diameter group command in the following tables is differentiated from a single-session related command by a preceding 'G' (Group). A Group Re-Auth Request, which applies to one or multiple session groups, has been exemplarily described in Section 6.1. Such Group RAR command is denoted as 'GRAR' in the following table. The same notation applies to other commands as per [RFC6733].

CLIENT, STATEFUL				
State	Event	Action	New State	
Idle	Client or Device Requests access	Send service specific auth req optionally including groups	Pending	

Open	GASR received with Group-Response-Action = ALL_GROUPS, session is assigned to received group(s) and client will comply with request to end the session	Send GASA with Result-Code = SUCCESS, Send GSTR.	Discon
Open	GASR received with Group-Response-Action = PER_GROUPS, session is assigned to received group(s) and client will comply with request to end the session	Send GASA with Result-Code = SUCCESS, Send GSTR per group	Discon
Open	GASR received with Group-Response-Action = PER_SESSION, session is assigned to received group(s) and client will comply with request to end the session	Send GASA with Result-Code = SUCCESS, Send STR per session	Discon
Open	GASR received, client will not comply with request to end all session in received group(s)	Send GASA with Result-Code != SUCCESS	Open
Discon	GSTA Received	Discon. user/device	Idle
Open	GRAR received with Group-Response-Action = ALL_GROUPS, session is assigned to received group(s) and client will perform subsequent re-auth	Send GRAA, Send service specific group re-auth req	Pending
Open	GRAR received with Group-Response-Action = PER_GROUP, session is assigned to received group(s) and client will perform subsequent re-auth	Send GRAA, Send service specific group re-auth req per group	Pending
Open	GRAR received with	Send GRAA,	Pending

	Group-Response-Action = PER_SESSION, session is assigned to received group(s) and client will perform subsequent re-auth	Send service specific re-auth req per session	
Open	GRAR received and client will not perform subsequent re-auth	Send GRAA with Result-Code != SUCCESS, Discon. user/device	Idle
Pending	Successful service-specific group re-authorization answer received.	Provide service	Open
Pending	Failed service-specific group re-authorization answer received.	Discon. user/device	Idle

The following state machine is observed by a server when it is maintaining state for the session. State transitions which are unmodified from [RFC6733] are not repeated here.

SERVER, STATEFUL				
State	Event	Action	New State	

Idle	Service-specific authorization request received, and user is authorized	Send successful service specific answer optionally including groups	Open	
Open	Server wants to terminate group(s)	Send GASR	Discon	
Discon	GASA received	Cleanup	Idle	
Any	GSTR received	Send GSTA, Cleanup	Idle	
Open	Server wants to reauth group(s)	Send GRAR	Pending	
Pending	GRAA received with Result-Code = SUCCESS	Update session(s)	Open	
Pending	GRAA received with Result-Code != SUCCESS	Cleanup session(s)	Idle	
Open	Service-specific group re-authoization request received and user is authorized	Send successful service specific group re-auth answer	Open	
Open	Service-specific group re-authorization request received and user is not authorized	Send failed service specific group re-auth answer, cleanup	Idle	

Authors' Addresses

Mark Jones

Email: mark@azu.ca

Marco Liebsch

Email: marco.liebsch@neclab.eu

Lionel Morand

Email: lionel.morand@orange.com

Network Working Group
Internet-Draft
Intended status: Informational
Expires: April 3, 2014

E. McMurry
B. Campbell
Tekelec
September 30, 2013

Diameter Overload Control Requirements
draft-ietf-dime-overload-reqs-13

Abstract

When a Diameter server or agent becomes overloaded, it needs to be able to gracefully reduce its load, typically by informing clients to reduce sending traffic for some period of time. Otherwise, it must continue to expend resources parsing and responding to Diameter messages, possibly resulting in a progressively more severe overload condition. The existing Diameter mechanisms are not sufficient for this purpose. This document describes the limitations of the existing mechanisms. Requirements for new overload management mechanisms are also provided.

Status of this Memo

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

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

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

This Internet-Draft will expire on April 3, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect

to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
1.1. Documentation Conventions	3
1.2. Causes of Overload	4
1.3. Effects of Overload	5
1.4. Overload vs. Network Congestion	5
1.5. Diameter Applications in a Broader Network	6
2. Overload Control Scenarios	6
2.1. Peer to Peer Scenarios	7
2.2. Agent Scenarios	9
2.3. Interconnect Scenario	12
3. Diameter Overload Case Studies	13
3.1. Overload in Mobile Data Networks	13
3.2. 3GPP Study on Core Network Overload	15
4. Existing Mechanisms	15
5. Issues with the Current Mechanisms	16
5.1. Problems with Implicit Mechanism	17
5.2. Problems with Explicit Mechanisms	17
6. Extensibility and Application Independence	18
7. Solution Requirements	19
7.1. General	19
7.2. Performance	20
7.3. Heterogeneous Support for Solution	21
7.4. Granular Control	21
7.5. Priority and Policy	22
7.6. Security	22
7.7. Flexibility and Extensibility	23
8. IANA Considerations	24
9. Security Considerations	24
9.1. Access Control	24
9.2. Denial-of-Service Attacks	25
9.3. Replay Attacks	25
9.4. Man-in-the-Middle Attacks	25
9.5. Compromised Hosts	26
10. References	26
10.1. Normative References	26
10.2. Informative References	26
Appendix A. Contributors	27
Appendix B. Acknowledgements	27
Authors' Addresses	28

1. Introduction

A Diameter [RFC6733] node is said to be overloaded when it has insufficient resources to successfully process all of the Diameter requests that it receives. When a node becomes overloaded, it needs to be able to gracefully reduce its load, typically by informing clients to reduce sending traffic for some period of time. Otherwise, it must continue to expend resources parsing and responding to Diameter messages, possibly resulting in a progressively more severe overload condition. The existing mechanisms provided by Diameter are not sufficient for this purpose. This document describes the limitations of the existing mechanisms, and provides requirements for new overload management mechanisms.

This document draws on the work done on SIP overload control ([RFC5390], [RFC6357]) as well as on experience gained via overload handling in Signaling System No. 7 (SS7) networks and studies done by the Third Generation Partnership Project (3GPP) (Section 3).

Diameter is not typically an end-user protocol; rather it is generally used as one component in support of some end-user activity.

For example, a SIP server might use Diameter to authenticate and authorize user access. Overload in the Diameter backend infrastructure will likely impact the experience observed by the end user in the SIP application.

The impact of Diameter overload on the client application (a client application may use the Diameter protocol and other protocols to do its job) is beyond the scope of this document.

This document presents non-normative descriptions of causes of overload along with related scenarios and studies. Finally, it offers a set of normative requirements for an improved overload indication mechanism.

1.1. Documentation Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as defined in [RFC2119], with the exception that they are not intended for interoperability of implementations. Rather, they are used to describe requirements towards future specifications where the interoperability requirements will be defined.

The terms "client", "server", "agent", "node", "peer", "upstream", and "downstream" are used as defined in [RFC6733].

1.2. Causes of Overload

Overload occurs when an element, such as a Diameter server or agent, has insufficient resources to successfully process all of the traffic it is receiving. Resources include all of the capabilities of the element used to process a request, including CPU processing, memory, I/O, and disk resources. It can also include external resources such as a database or DNS server, in which case the CPU, processing, memory, I/O, and disk resources of those elements are effectively part of the logical element processing the request.

External resources can include upstream Diameter nodes; for example, a Diameter agent can become effectively overloaded if one or more upstream nodes are overloaded.

A Diameter node can become overloaded due to request levels that exceed its capacity, a reduction of available resources (for example, a local or upstream hardware failure) or a combination of the two.

Overload can occur for many reasons, including:

Inadequate capacity: When designing Diameter networks, that is, application layer multi-node Diameter deployments, it can be very difficult to predict all scenarios that may cause elevated traffic. It may also be more costly to implement support for some scenarios than a network operator may deem worthwhile. This results in the likelihood that a Diameter network will not have adequate capacity to handle all situations.

Dependency failures: A Diameter node can become overloaded because a resource on which it depends has failed or become overloaded, greatly reducing the logical capacity of the node. In these cases, even minimal traffic might cause the node to go into overload. Examples of such dependency overloads include DNS servers, databases, disks, and network interfaces.

Component failures: A Diameter node can become overloaded when it is a member of a cluster of servers that each share the load of traffic, and one or more of the other members in the cluster fail. In this case, the remaining nodes take over the work of the failed nodes. Normally, capacity planning takes such failures into account, and servers are typically run with enough spare capacity to handle failure of another node. However, unusual failure conditions can cause many nodes to fail at once. This is often the case with software failures, where a bad packet or bad database entry hits the same bug in a set of nodes in a cluster.

Network Initiated Traffic Flood: Issues with the radio access network in a mobile network such as radio overlays with frequent handovers, and operational changes are examples of network events that can precipitate a flood of Diameter signaling traffic, such as an avalanche restart. Failure of a Diameter proxy may also result in a large amount of signaling as connections and sessions are reestablished.

Subscriber Initiated Traffic Flood: Large gatherings of subscribers or events that result in many subscribers interacting with the network in close time proximity can result in Diameter signaling traffic floods. For example, the finale of a large fireworks show could be immediately followed by many subscribers posting messages, pictures, and videos concentrated on one portion of a network. Subscriber devices, such as smartphones, may use aggressive registration strategies that generate unusually high Diameter traffic loads.

DoS attacks: An attacker, wishing to disrupt service in the network, can cause a large amount of traffic to be launched at a target element. This can be done from a central source of traffic or through a distributed DoS attack. In all cases, the volume of traffic well exceeds the capacity of the element, sending the system into overload.

1.3. Effects of Overload

Modern Diameter networks, composed of application layer multi-node deployments of Diameter elements, may operate at very large transaction volumes. If a Diameter node becomes overloaded, or even worse, fails completely, a large number of messages may be lost very quickly. Even with redundant servers, many messages can be lost in the time it takes for failover to complete. While a Diameter client or agent should be able to retry such requests, an overloaded peer may cause a sudden large increase in the number of transaction transactions needing to be retried, rapidly filling local queues or otherwise contributing to local overload. Therefore Diameter devices need to be able to shed load before critical failures can occur.

1.4. Overload vs. Network Congestion

This document uses the term "overload" to refer to application-layer overload at Diameter nodes. This is distinct from "network congestion", that is, congestion that occurs at the lower networking layers that may impact the delivery of Diameter messages between nodes. This document recognizes that element overload and network congestion are interrelated, and that overload can contribute to network congestion and vice versa.

Network congestion issues are better handled by the transport protocols. Diameter uses TCP and SCTP, both of which include congestion management features. Analysis of whether those features are sufficient for transport level congestion between Diameter nodes, and any work to further mitigate network congestion is out of scope both for this document, and for the work proposed by this document.

1.5. Diameter Applications in a Broader Network

Most elements using Diameter applications do not use Diameter exclusively. It is important to realize that overload of an element can be caused by a number of factors that may be unrelated to the processing of Diameter or Diameter applications.

An element that doesn't use Diameter exclusively needs to be able to signal to Diameter peers that it is experiencing overload regardless of the cause of the overload, since the overload will affect that element's ability to process Diameter transactions. If the element communicates with protocols other than Diameter, it may also need to signal the overload situation on these protocols depending on its function and the architecture of the network and application it is providing services for. Whether that is necessary can only be decided within the context of that architecture and use cases. A mechanism for signaling overload with Diameter, which this specification details the requirements for, provides Diameter nodes the ability to signal their Diameter peers of overload, mitigating that part of the issue. Diameter nodes may need to use this, as well as other mechanisms, to solve their broader overload issues. Indicating overload on protocols other than Diameter is out of scope for this document, and for the work proposed by this document.

2. Overload Control Scenarios

Several Diameter deployment scenarios exist that may impact overload management. The following scenarios help motivate the requirements for an overload management mechanism.

These scenarios are by no means exhaustive, and are in general simplified for the sake of clarity. In particular, this document assumes for the sake of clarity that the client sends Diameter requests to the server, and the server sends responses to client, even though Diameter supports bidirectional applications. Each direction in such an application can be modeled separately.

In a large scale deployment, many of the nodes represented in these scenarios would be deployed as clusters of servers. This document assumes that such a cluster is responsible for managing its own

internal load balancing and overload management so that it appears as a single Diameter node. That is, other Diameter nodes can treat it as single, monolithic node for the purposes of overload management.

These scenarios do not illustrate the client application. As mentioned in Section 1, Diameter is not typically an end-user protocol; rather it is generally used in support of some other client application. These scenarios do not consider the impact of Diameter overload on the client application.

2.1. Peer to Peer Scenarios

This section describes Diameter peer-to-peer scenarios. That is, scenarios where a Diameter client talks directly with a Diameter server, without the use of a Diameter agent.

Figure 1 illustrates the simplest possible Diameter relationship. The client and server share a one-to-one peer-to-peer relationship. If the server becomes overloaded, either because the client exceeds the server's capacity, or because the server's capacity is reduced due to some resource dependency, the client needs to reduce the amount of Diameter traffic it sends to the server. Since the client cannot forward requests to another server, it must either queue requests until the server recovers, or itself become overloaded in the context of the client application and other protocols it may also use.

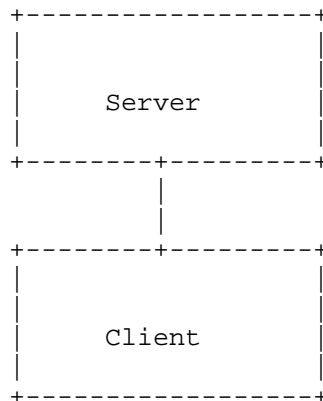


Figure 1: Basic Peer to Peer Scenario

Figure 2 shows a similar scenario, except in this case the client has multiple servers that can handle work for a specific realm and

application. If server 1 becomes overloaded, the client can forward traffic to server 2. Assuming server 2 has sufficient reserve capacity to handle the forwarded traffic, the client should be able to continue serving client application protocol users. If server 1 is approaching overload, but can still handle some number of new request, it needs to be able to instruct the client to forward a subset of its traffic to server 2.

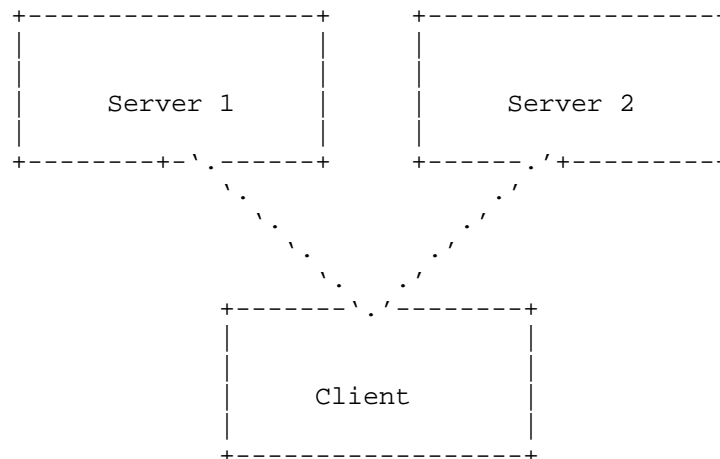


Figure 2: Multiple Server Peer to Peer Scenario

Figure 3 illustrates a peer-to-peer scenario with multiple Diameter realm and application combinations. In this example, server 2 can handle work for both applications. Each application might have different resource dependencies. For example, a server might need to access one database for application A, and another for application B. This creates a possibility that Server 2 could become overloaded for application A but not for application B, in which case the client would need to divert some part of its application A requests to server 1, but should not divert any application B requests. This requires server 2 to be able to distinguish between applications when it indicates an overload condition to the client.

On the other hand, it's possible that the servers host many applications. If server 2 becomes overloaded for all applications, it would be undesirable for it to have to notify the client separately for each application. Therefore it also needs a way to indicate that it is overloaded for all possible applications.

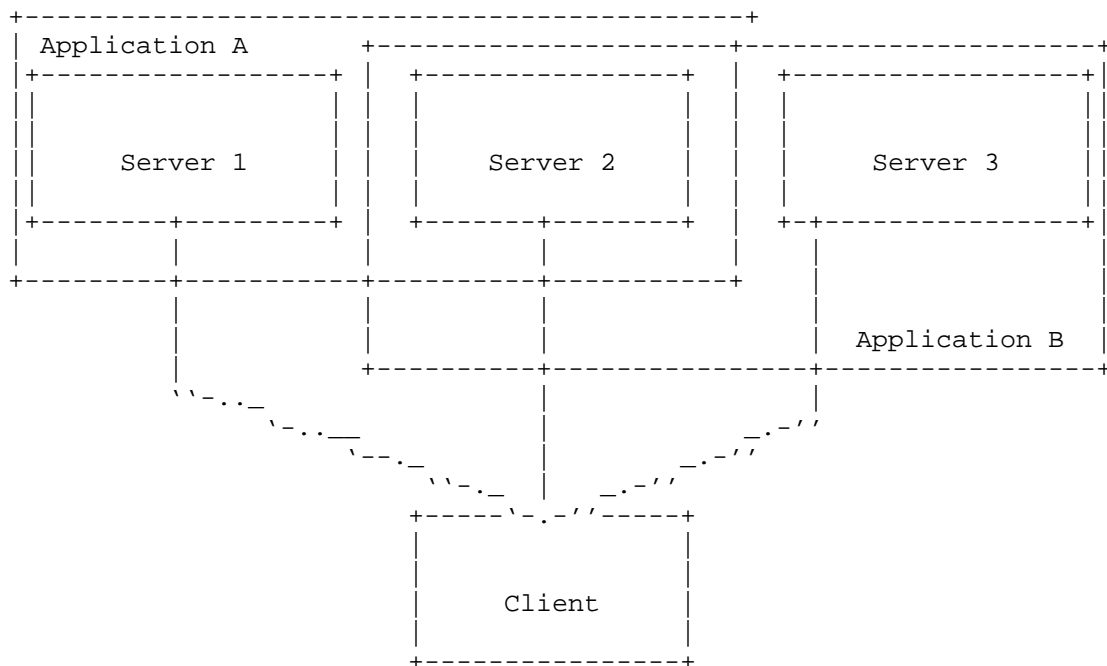


Figure 3: Multiple Application Peer to Peer Scenario

2.2. Agent Scenarios

This section describes scenarios that include a Diameter agent, either in the form of a Diameter relay or Diameter proxy. These scenarios do not consider Diameter redirect agents, since they are more readily modeled as end-servers. The examples have been kept simple deliberately, to illustrate basic concepts. Significantly more complicated topologies are possible with Diameter, including multiple intermediate agents in a path connected in a variety of ways.

Figure 4 illustrates a simple Diameter agent scenario with a single client, agent, and server. In this case, overload can occur at the server, at the agent, or both. But in most cases, client behavior is the same whether overload occurs at the server or at the agent. From the client's perspective, server overload and agent overload is the same thing.

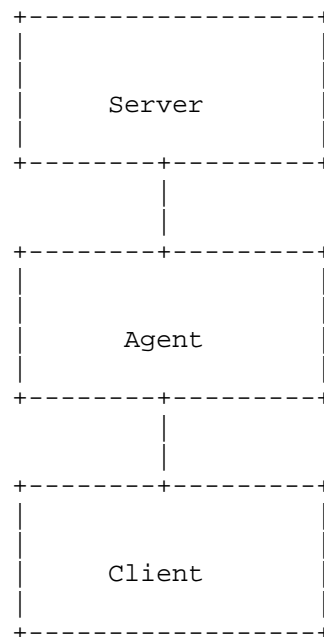


Figure 4: Basic Agent Scenario

Figure 5 shows an agent scenario with multiple servers. If server 1 becomes overloaded, but server 2 has sufficient reserve capacity, the agent may be able to transparently divert some or all Diameter requests originally bound for server 1 to server 2.

In most cases, the client does not have detailed knowledge of the Diameter topology upstream of the agent. If the agent uses dynamic discovery to find eligible servers, the set of eligible servers may not be enumerable from the perspective of the client. Therefore, in most cases the agent needs to deal with any upstream overload issues in a way that is transparent to the client. If one server notifies the agent that it has become overloaded, the notification should not be passed back to the client in a way that the client could mistakenly perceive the agent itself as being overloaded. If the set of all possible destinations upstream of the agent no longer has sufficient capacity for incoming load, the agent itself becomes effectively overloaded.

On the other hand, there are cases where the client needs to be able to select a particular server from behind an agent. For example, if a Diameter request is part of a multiple-round-trip authentication, or is otherwise part of a Diameter "session", it may have a

DestinationHost AVP that requires the request to be served by server 1. Therefore the agent may need to inform a client that a particular upstream server is overloaded or otherwise unavailable. Note that there can be many ways a server can be specified, which may have different implications (e.g. by IP address, by host name, etc).

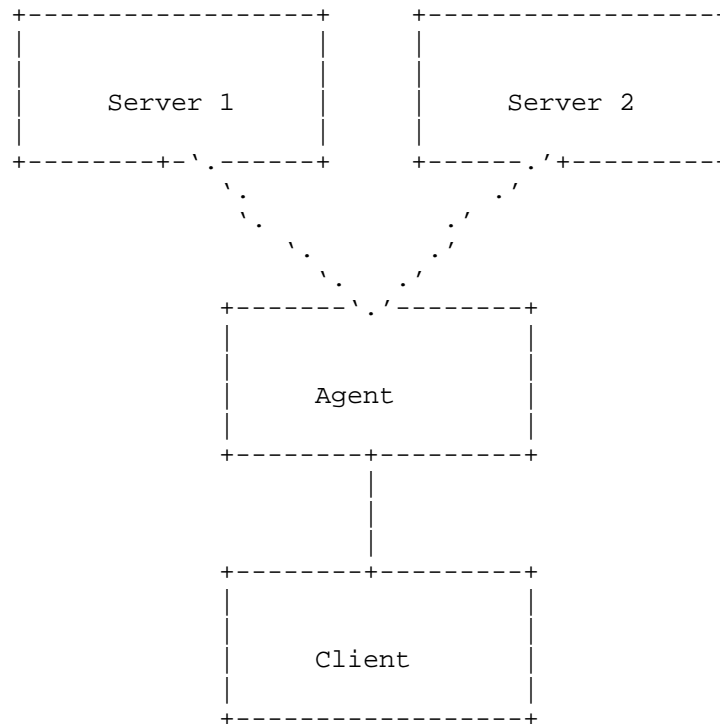


Figure 5: Multiple Server Agent Scenario

Figure 6 shows a scenario where an agent routes requests to a set of servers for more than one Diameter realm and application. In this scenario, if server 1 becomes overloaded or unavailable while server 2 still has available capacity, the agent may effectively operate at reduced capacity for application A, but at full capacity for application B. Therefore, the agent needs to be able to report that it is overloaded for one application, but not for another.

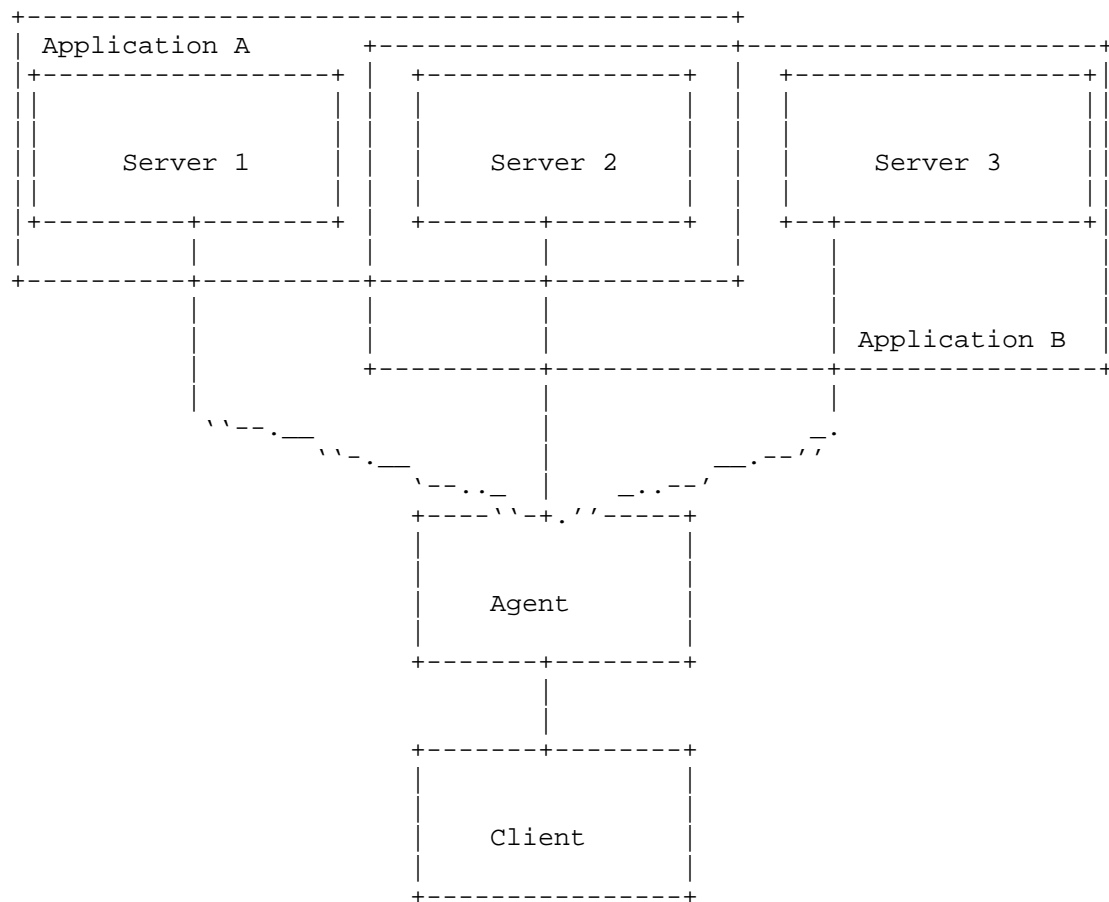


Figure 6: Multiple Application Agent Scenario

2.3. Interconnect Scenario

Another scenario to consider when looking at Diameter overload is that of multiple network operators using Diameter components connected through an interconnect service, e.g. using IPX. IPX (IP eXchange) [IR.34] is an Inter-Operator IP Backbone that provides roaming interconnection network between mobile operators and service providers. The IPX is also used to transport Diameter signaling between operators [IR.88]. Figure 7 shows two network operators with an interconnect network in-between. There could be any number of these networks between any two network operator's networks.

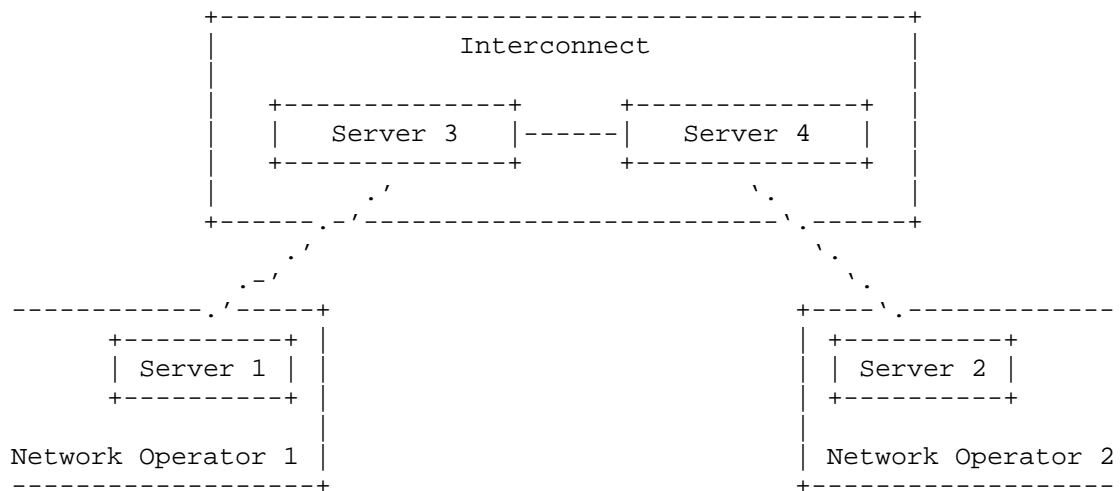


Figure 7: Two Network Interconnect Scenario

The characteristics of the information that an operator would want to share over such a connection are different from the information shared between components within a network operator's network. Network operators may not want to convey topology or operational information, which limits how much overload and loading information can be sent. For the interconnect scenario shown, Server 2 may want to signal overload to Server 1, to affect traffic coming from Network Operator 1.

This case is distinct from those internal to a network operator's network, where there may be many more elements in a more complicated topology. Also, the elements in the interconnect network may not support Diameter overload control, and the network operators may not want the interconnect network to use overload or loading information. They may only want the information to pass through the interconnect network without further processing or action by the interconnect network even if the elements in the interconnect network do support Diameter overload control.

3. Diameter Overload Case Studies

3.1. Overload in Mobile Data Networks

As the number of Third Generation (3G) and Long Term Evolution (LTE) enabled smartphone devices continue to expand in mobile networks, there have been situations where high signaling traffic load led to overload events at the Diameter-based Home Location Registries (HLR)

and/or Home Subscriber Servers (HSS) [TR23.843]. The root causes of the HLR overload events were manifold but included hardware failure and procedural errors. The result was high signaling traffic load on the HLR and HSS.

The 3GPP architecture [TS23.002] makes extensive use of Diameter. It is used for mobility management [TS29.272] (and others), (IP Multimedia Subsystem) IMS [TS29.228] (and others), policy and charging control [TS29.212] (and others) as well as other functions. The details of the architecture are out of scope for this document, but it is worth noting that there are quite a few Diameter applications, some with quite large amounts of Diameter signaling in deployed networks.

The 3GPP specifications do not currently address overload for Diameter applications or provide an equivalent load control mechanism to those provided in the more traditional SS7 elements in (Global System for Mobile Communications) GSM [TS29.002]. The capabilities specified in the 3GPP standards do not adequately address the abnormal condition where excessively high signaling traffic load situations are experienced.

Smartphones, an increasingly large percentage of mobile devices, contribute much more heavily, relative to non-smartphones, to the continuation of a registration surge due to their very aggressive registration algorithms. Smartphone behavior contributes to network loading and can contribute to overload conditions. The aggressive smartphone logic is designed to:

- a. always have voice and data registration, and
- b. constantly try to be on 3G or LTE data (and thus on 3G voice or VoLTE [IR.92]) for their added benefits.

Non-smartphones typically have logic to wait for a time period after registering successfully on voice and data.

The smartphone aggressive registration is problematic in two ways:

- o first by generating excessive signaling load towards the HSS that is ten times that from a non-smartphone,
- o and second by causing continual registration attempts when a network failure affects registrations through the 3G data network.

3.2. 3GPP Study on Core Network Overload

A study in 3GPP SA2 on core network overload has produced the technical report [TR23.843]. This enumerates several causes of overload in mobile core networks including portions that are signaled using Diameter. TR23.843 is a work in progress and is not complete. However, it is useful for pointing out scenarios and the general need for an overload control mechanism for Diameter.

It is common for mobile networks to employ more than one radio technology and to do so in an overlay fashion with multiple technologies present in the same location (such as 2nd or 3rd generation mobile technologies along with LTE). This presents opportunities for traffic storms when issues occur on one overlay and not another as all devices that had been on the overlay with issues switch. This causes a large amount of Diameter traffic as locations and policies are updated.

Another scenario called out by this study is a flood of registration and mobility management events caused by some element in the core network failing. This flood of traffic from end nodes falls under the network initiated traffic flood category. There is likely to also be traffic resulting directly from the component failure in this case. A similar flood can occur when elements or components recover as well.

Subscriber initiated traffic floods are also indicated in this study as an overload mechanism where a large number of mobile devices attempting to access services at the same time, such as in response to an entertainment event or a catastrophic event.

While this 3GPP study is concerned with the broader effects of these scenarios on wireless networks and their elements, they have implications specifically for Diameter signaling. One of the goals of this document is to provide guidance for a core mechanism that can be used to mitigate the scenarios called out by this study.

4. Existing Mechanisms

Diameter offers both implicit and explicit mechanisms for a Diameter node to learn that a peer is overloaded or unreachable. The implicit mechanism is simply the lack of responses to requests. If a client fails to receive a response in a certain time period, it assumes the upstream peer is unavailable, or overloaded to the point of effective unavailability. The watchdog mechanism [RFC3539] ensures that a certain rate of transaction responses occur even when there is otherwise little or no other Diameter traffic.

The explicit mechanism can involve specific protocol error responses, where an agent or server tells a downstream peer that it is either too busy to handle a request (`DIAMETER_TOO_BUSY`) or unable to route a request to an upstream destination (`DIAMETER_UNABLE_TO_DELIVER`), perhaps because that destination itself is overloaded to the point of unavailability.

Another explicit mechanism, a DPR (Disconnect-Peer-Request) message, can be sent with a Disconnect-Cause of `BUSY`. This signals the sender's intent to close the transport connection, and requests the client not to reconnect.

Once a Diameter node learns that an upstream peer has become overloaded via one of these mechanisms, it can then attempt to take action to reduce the load. This usually means forwarding traffic to an alternate destination, if available. If no alternate destination is available, the node must either reduce the number of messages it originates (in the case of a client) or inform the client to reduce traffic (in the case of an agent.)

Diameter requires the use of a congestion-managed transport layer, currently TCP or SCTP, to mitigate network congestion. It is expected that these transports manage network congestion and that issues with transport (e.g. congestion propagation and window management) are managed at that level. But even with a congestion-managed transport, a Diameter node can become overloaded at the Diameter protocol or application layers due to the causes described in Section 1.2 and congestion managed transports do not provide facilities (and are at the wrong level) to handle server overload. Transport level congestion management is also not sufficient to address overload in cases of multi-hop and multi-destination signaling.

5. Issues with the Current Mechanisms

The currently available Diameter mechanisms for indicating an overload condition are not adequate to avoid service outages due to overload. This inadequacy may, in turn, contribute to broader impacts resulting from overload due to unresponsive Diameter nodes causing application or transport layer retransmissions. In particular, they do not allow a Diameter agent or server to shed load as it approaches overload. At best, a node can only indicate that it needs to entirely stop receiving requests, i.e. that it has effectively failed. Even that is problematic due to the inability to indicate durational validity on the transient errors available in the base Diameter protocol. Diameter offers no mechanism to allow a node to indicate different overload states for different categories of

messages, for example, if it is overloaded for one Diameter application but not another.

5.1. Problems with Implicit Mechanism

The implicit mechanism doesn't allow an agent or server to inform the client of a problem until it is effectively too late to do anything about it. The client does not know to take action until the upstream node has effectively failed. A Diameter node has no opportunity to shed load early to avoid collapse in the first place.

Additionally, the implicit mechanism cannot distinguish between overload of a Diameter node and network congestion. Diameter treats the failure to receive an answer as a transport failure.

5.2. Problems with Explicit Mechanisms

The Diameter specification is ambiguous on how a client should handle receipt of a `DIAMETER_TOO_BUSY` response. The base specification [RFC6733] indicates that the sending client should attempt to send the request to a different peer. It makes no suggestion that the receipt of a `DIAMETER_TOO_BUSY` response should affect future Diameter messages in any way.

The Authentication, Authorization, and Accounting (AAA) Transport Profile [RFC3539] recommends that a AAA node that receives a "Busy" response failover all remaining requests to a different agent or server. But while the Diameter base specification explicitly depends on RFC3539 to define transport behavior, it does not refer to RFC3539 in the description of behavior on receipt of `DIAMETER_TOO_BUSY`. There's a strong likelihood that at least some implementations will continue to send Diameter requests to an upstream peer even after receiving a `DIAMETER_TOO_BUSY` error.

BCP 41 [RFC2914] describes, among other things, how end-to-end application behavior can help avoid congestion collapse. In particular, an application should avoid sending messages that will never be delivered or processed. The `DIAMETER_TOO_BUSY` behavior as described in the Diameter base specification fails at this, since if an upstream node becomes overloaded, a client attempts each request, and does not discover the need to failover the request until the initial attempt fails.

The situation is improved if implementations follow the [RFC3539] recommendation and keep state about upstream peer overload. But even then, the Diameter specification offers no guidance on how long a client should wait before retrying the overloaded destination. If an agent or server supports multiple realms and/or applications,

DIAMETER_TOO_BUSY offers no way to indicate that it is overloaded for one application but not another. A DIAMETER_TOO_BUSY error can only indicate overload at a "whole server" scope.

Agent processing of a DIAMETER_TOO_BUSY response is also problematic as described in the base specification. DIAMETER_TOO_BUSY is defined as a protocol error. If an agent receives a protocol error, it may either handle it locally or it may forward the response back towards the downstream peer. If a downstream peer receives the DIAMETER_TOO_BUSY response, it may stop sending all requests to the agent for some period of time, even though the agent may still be able to deliver requests to other upstream peers.

DIAMETER_UNABLE_TO_DELIVER, or using DPR with cause code BUSY also have no mechanisms for specifying the scope or cause of the failure, or the durational validity.

The issues with error responses in [RFC6733] extend beyond the particular issues for overload control and have been addressed in an ad hoc fashion by various implementations. Addressing these in a standard way would be a useful exercise, but it is beyond the scope of this document.

6. Extensibility and Application Independence

Given the variety of scenarios Diameter elements can be deployed in, and the variety of roles they can fulfill with Diameter and other technologies, a single algorithm for handling overload may not be sufficient. For purposes of this discussion, algorithm is inclusive of behavior for control of overload, but does not encompass the general mechanism or transport of control information. This effort cannot anticipate all possible future scenarios and roles. Extensibility, particularly of algorithms used to deal with overload, will be important to cover these cases.

Similarly, the scopes that overload information may apply to may include cases that have not yet been considered. Extensibility in this area will also be important.

The basic mechanism is intended to be application-independent, that is, a Diameter node can use it across any existing and future Diameter applications and expect reasonable results. Certain Diameter applications might, however, benefit from application-specific behavior over and above the mechanism's defaults. For example, an application specification might specify relative priorities of messages or selection of a specific overload control algorithm.

7. Solution Requirements

This section proposes requirements for an improved mechanism to control Diameter overload, with the goals of addressing the issues described in Section 5 and supporting the scenarios described in Section 2. These requirements are stated primarily in terms of individual node behavior to inform the design of the improved mechanism; solution designers should keep in mind that the overall goal is improved overall system behavior across all the nodes involved, not just improved behavior from specific individual nodes.

7.1. General

- REQ 1: The solution **MUST** provide a communication method for Diameter nodes to exchange load and overload information.
- REQ 2: The solution **MUST** allow Diameter nodes to support overload control regardless of which Diameter applications they support. Diameter clients and agents must be able to use the received load and overload information to support graceful behavior during an overload condition. Graceful behavior under overload conditions is best described by REQ 3.
- REQ 3: The solution **MUST** limit the impact of overload on the overall useful throughput of a Diameter server, even when the incoming load on the network is far in excess of its capacity. The overall useful throughput under load is the ultimate measure of the value of a solution.
- REQ 4: Diameter allows requests to be sent from either side of a connection and either side of a connection may have need to provide its overload status. The solution **MUST** allow each side of a connection to independently inform the other of its overload status.
- REQ 5: Diameter allows nodes to determine their peers via dynamic discovery or manual configuration. The solution **MUST** work consistently without regard to how peers are determined.
- REQ 6: The solution designers **SHOULD** seek to minimize the amount of new configuration required in order to work. For example, it is better to allow peers to advertise or negotiate support for the solution, rather than to require this knowledge to be configured at each node.

7.2. Performance

- REQ 7: The solution and any associated default algorithm(s) MUST ensure that the system remains stable. At some point after an overload condition has ended, the solution MUST enable capacity to stabilize and become equal to what it would be in the absence of an overload condition. Note that this also requires that the solution MUST allow nodes to shed load without introducing non converging oscillations during or after an overload condition.
- REQ 8: Supporting nodes MUST be able to distinguish current overload information from stale information.
- REQ 9: The solution MUST function across fully loaded as well as quiescent transport connections. This is partially derived from the requirement for stability in REQ 7.
- REQ 10: Consumers of overload information MUST be able to determine when the overload condition improves or ends.
- REQ 11: The solution MUST be able to operate in networks of different sizes.
- REQ 12: When a single network node fails, goes into overload, or suffers from reduced processing capacity, the solution MUST make it possible to limit the impact of this on other nodes in the network. This helps to prevent a small-scale failure from becoming a widespread outage.
- REQ 13: The solution MUST NOT introduce substantial additional work for node in an overloaded state. For example, a requirement for an overloaded node to send overload information every time it received a new request would introduce substantial work.
- REQ 14: Some scenarios that result in overload involve a rapid increase of traffic with little time between normal levels and overload inducing levels. The solution SHOULD provide for rapid feedback when traffic levels increase.
- REQ 15: The solution MUST NOT interfere with the congestion control mechanisms of underlying transport protocols. For example, a solution that opened additional TCP connections when the network is congested would reduce the effectiveness of the underlying congestion control mechanisms.

7.3. Heterogeneous Support for Solution

- REQ 16: The solution is likely to be deployed incrementally. The solution MUST support a mixed environment where some, but not all, nodes implement it.
- REQ 17: In a mixed environment with nodes that support the solution and that do not, the solution MUST NOT result in materially less useful throughput during overload as would have resulted if the solution were not present. It SHOULD result in less severe overload in this environment.
- REQ 18: In a mixed environment of nodes that support the solution and that do not, the solution MUST NOT preclude elements that support overload control from treating elements that do not support overload control in a equitable fashion relative to those that do. Users and operators of nodes that do not support the solution MUST NOT unfairly benefit from the solution. The solution specification SHOULD provide guidance to implementors for dealing with elements not supporting overload control.
- REQ 19: It MUST be possible to use the solution between nodes in different realms and in different administrative domains.
- REQ 20: Any explicit overload indication MUST be clearly distinguishable from other errors reported via Diameter.
- REQ 21: In cases where a network node fails, is so overloaded that it cannot process messages, or cannot communicate due to a network failure, it may not be able to provide explicit indications of the nature of the failure or its levels of overload. The solution MUST result in at least as much useful throughput as would have resulted if the solution was not in place.

7.4. Granular Control

- REQ 22: The solution MUST provide a way for a node to throttle the amount of traffic it receives from a peer node. This throttling SHOULD be graded so that it can be applied gradually as offered load increases. Overload is not a binary state; there may be degrees of overload.

- REQ 23: The solution MUST provide sufficient information to enable a load balancing node to divert messages that are rejected or otherwise throttled by an overloaded upstream node to other upstream nodes that are the most likely to have sufficient capacity to process them.
- REQ 24: The solution MUST provide a mechanism for indicating load levels even when not in an overloaded condition, to assist nodes making decisions to prevent overload conditions from occurring.

7.5. Priority and Policy

- REQ 25: The base specification for the solution SHOULD offer general guidance on which message types might be desirable to send or process over others during times of overload, based on application-specific considerations. For example, it may be more beneficial to process messages for existing sessions ahead of new sessions. Some networks may have a requirement to give priority to requests associated with emergency sessions. Any normative or otherwise detailed definition of the relative priorities of message types during an overload condition will be the responsibility of the application specification.
- REQ 26: The solution MUST NOT prevent a node from prioritizing requests based on any local policy, so that certain requests are given preferential treatment, given additional retransmission, not throttled, or processed ahead of others.

7.6. Security

- REQ 27: The solution MUST NOT provide new vulnerabilities to malicious attack, or increase the severity of any existing vulnerabilities. This includes vulnerabilities to DoS and DDoS attacks as well as replay and man-in-the middle attacks. Note that the Diameter base specification [RFC6733] lacks end to end security and this must be considered (see the Security Considerations (Section 9)). Note that this requirement was expressed at a high level so as to not preclude any particular solution. It is expected that the solution will address this in more detail.

- REQ 28: The solution MUST NOT depend on being deployed in environments where all Diameter nodes are completely trusted. It SHOULD operate as effectively as possible in environments where other nodes are malicious; this includes preventing malicious nodes from obtaining more than a fair share of service. Note that this does not imply any responsibility on the solution to detect, or take countermeasures against, malicious nodes.
- REQ 29: It MUST be possible for a supporting node to make authorization decisions about what information will be sent to peer nodes based on the identity of those nodes. This allows a domain administrator who considers the load of their nodes to be sensitive information to restrict access to that information. Of course, in such cases, there is no expectation that the solution itself will help prevent overload from that peer node.
- REQ 30: The solution MUST NOT interfere with any Diameter compliant method that a node may use to protect itself from overload from non-supporting nodes, or from denial of service attacks.

7.7. Flexibility and Extensibility

- REQ 31: There are multiple situations where a Diameter node may be overloaded for some purposes but not others. For example, this can happen to an agent or server that supports multiple applications, or when a server depends on multiple external resources, some of which may become overloaded while others are fully available. The solution MUST allow Diameter nodes to indicate overload with sufficient granularity to allow clients to take action based on the overloaded resources without unreasonably forcing available capacity to go unused. The solution MUST support specification of overload information with granularities of at least "Diameter node", "realm", and "Diameter application", and MUST allow extensibility for others to be added in the future.
- REQ 32: The solution MUST provide a method for extending the information communicated and the algorithms used for overload control.

REQ 33: The solution MUST provide a default algorithm that is mandatory to implement.

REQ 34: The solution SHOULD provide a method for exchanging overload and load information between elements that are connected by intermediaries that do not support the solution.

8. IANA Considerations

This document makes no requests of IANA.

9. Security Considerations

A Diameter overload control mechanism is primarily concerned with the load and overload related behavior of nodes in a Diameter network, and the information used to affect that behavior. Load and overload information is shared between nodes and directly affects the behavior and thus is potentially vulnerable to a number of methods of attack.

Load and overload information may also be sensitive from both business and network protection viewpoints. Operators of Diameter equipment want to control visibility to load and overload information to keep it from being used for competitive intelligence or for targeting attacks. It is also important that the Diameter overload control mechanism not introduce any way in which any other information carried by Diameter is sent inappropriately.

Note that the Diameter base specification [RFC6733] lacks end to end security, making verifying the authenticity and ownership of load and overload information difficult for non-adjacent nodes. Authentication of load and overload information helps to alleviate several of the security issues listed in this section.

This document includes requirements intended to mitigate the effects of attacks and to protect the information used by the mechanism. This section discusses potential security considerations for overload control solutions. This discussion provides the motivation for several normative requirements described in Section 7. The discussion includes specific references to the normative requirements that apply for each issue.

9.1. Access Control

To control the visibility of load and overload information, sending should be subject to some form of authentication and authorization of

the receiver. It is also important to the receivers that they are confident the load and overload information they receive is from a legitimate source. REQ 28 requires the solution to work without assuming that all Diameter nodes in a network are trusted for the purposes of exchanging overload and load information. REQ 29 requires the solution to let nodes restrict unauthorized parties from seeing overload information. Note that this implies a certain amount of configurability on the nodes supporting the Diameter overload control mechanism.

9.2. Denial-of-Service Attacks

An overload control mechanism provides a very attractive target for denial-of-service attacks. A small number of messages may affect a large service disruption by falsely reporting overload conditions. Alternately, attacking servers nearing, or in, overload may also be facilitated by disrupting their overload indications, potentially preventing them from mitigating their overload condition.

A design goal for the Diameter overload control mechanism is to minimize or eliminate the possibility of using the mechanism for this type of attack. More strongly, REQ 27 forbids the solution from introducing new vulnerabilities to malicious attack. Additionally, REQ 30 stipulates that the solution not interfere with other mechanisms used for protection against denial of service attacks.

As the intent of some denial-of-service attacks is to induce overload conditions, an effective overload control mechanism should help to mitigate the effects of an such an attack.

9.3. Replay Attacks

An attacker that has managed to obtain some messages from the overload control mechanism may attempt to affect the behavior of nodes supporting the mechanism by sending those messages at potentially inopportune times. In addition to time shifting, replay attacks may send messages to other nodes as well (target shifting).

A design goal for the Diameter overload control solution is to minimize or eliminate the possibility of causing disruption by using a replay attack on the Diameter overload control mechanism. (Allowing a replay attack using the overload control solution would violate REQ 27.)

9.4. Man-in-the-Middle Attacks

By inserting themselves in between two nodes supporting the Diameter overload control mechanism, an attacker may potentially both access

and alter the information sent between those nodes. This can be used for information gathering for business intelligence and attack targeting, as well as direct attacks.

REQs 27, 28, and 29 imply a need to prevent man-in-the-middle attacks on the overload control solution. A transport using TLS and/or IPSEC may be desirable for this purpose.

9.5. Compromised Hosts

A compromised host that supports the Diameter overload control mechanism could be used for information gathering as well as for sending malicious information to any Diameter node that would normally accept information from it. While it is beyond the scope of the Diameter overload control mechanism to mitigate any operational interruption to the compromised host, REQs 28 and 29 imply a need to minimize the impact that a compromised host can have on other nodes through the use of the Diameter overload control mechanism. Of course, a compromised host could be used to cause damage in a number of other ways. This is out of scope for a Diameter overload control mechanism.

10. References

10.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.
- [RFC2914] Floyd, S., "Congestion Control Principles", BCP 41, RFC 2914, September 2000.
- [RFC3539] Aboba, B. and J. Wood, "Authentication, Authorization and Accounting (AAA) Transport Profile", RFC 3539, June 2003.

10.2. Informative References

- [RFC5390] Rosenberg, J., "Requirements for Management of Overload in the Session Initiation Protocol", RFC 5390, December 2008.
- [RFC6357] Hilt, V., Noel, E., Shen, C., and A. Abdelal, "Design Considerations for Session Initiation Protocol (SIP) Overload Control", RFC 6357, August 2011.

- [TR23.843] 3GPP, "Study on Core Network Overload Solutions (Work in Progress)", TR 23.843 0.6.0, October 2012.
- [IR.34] GSMA, "Inter-Service Provider IP Backbone Guidelines", IR 34 7.0, January 2012.
- [IR.88] GSMA, "LTE Roaming Guidelines", IR 88 7.0, January 2012.
- [IR.92] GSMA, "IMS Profile for Voice and SMS", IR 92 7.0, March 2013.
- [TS23.002] 3GPP, "Network Architecture", TS 23.002 12.0.0, September 2012.
- [TS29.272] 3GPP, "Evolved Packet System (EPS); Mobility Management Entity (MME) and Serving GPRS Support Node (SGSN) related interfaces based on Diameter protocol", TS 29.272 11.4.0, September 2012.
- [TS29.212] 3GPP, "Policy and Charging Control (PCC) over Gx/Sd reference point", TS 29.212 11.6.0, September 2012.
- [TS29.228] 3GPP, "IP Multimedia (IM) Subsystem Cx and Dx interfaces; Signalling flows and message contents", TS 29.228 11.5.0, September 2012.
- [TS29.002] 3GPP, "Mobile Application Part (MAP) specification", TS 29.002 11.4.0, September 2012.

Appendix A. Contributors

Significant contributions to this document were made by Adam Roach and Eric Noel.

Appendix B. Acknowledgements

Review of, and contributions to, this specification by Martin Dolly, Carolyn Johnson, Jianrong Wang, Imtiaz Shaikh, Jouni Korhonen, Robert Sparks, Dieter Jacobsohn, Janet Gunn, Jean-Jacques Trottin, Laurent Thiebaut, Andrew Booth, and Lionel Morand were most appreciated. We

would like to thank them for their time and expertise.

Authors' Addresses

Eric McMurry
Tekelec
17210 Campbell Rd.
Suite 250
Dallas, TX 75252
US

Email: emcmurphy@computer.org

Ben Campbell
Tekelec
17210 Campbell Rd.
Suite 250
Dallas, TX 75252
US

Email: ben@nostrum.com

Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: February 3, 2013

G. Zorn
Network Zen
Q. Wu
Huawei
J. Korhonen
NSN
August 2, 2012

Diameter Support for Proxy Mobile IPv6 Localized Routing
draft-ietf-dime-pmip6-lr-18

Abstract

In Proxy Mobile IPv6, packets received from a Mobile Node (MN) by the Mobile Access Gateway (MAG) to which it is attached are typically tunneled to a Local Mobility Anchor (LMA) for routing. The term "localized routing" refers to a method by which packets are routed directly between an MN's MAG and the MAG of its Correspondent Node (CN) without involving any LMA. In a Proxy Mobile IPv6 deployment, it may be desirable to control the establishment of localized routing sessions between two MAGs in a Proxy Mobile IPv6 domain by requiring that the session be authorized. This document specifies how to accomplish this using the Diameter protocol.

Status of this Memo

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

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

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

This Internet-Draft will expire on February 3, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal

Provisions Relating to IETF Documents
(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	3
3. Solution Overview	3
4. Attribute Value Pair Used in this Document	4
4.1. User-Name AVP	5
4.2. PMIP6-IPv4-Home-Address AVP	5
4.3. MIP6-Home-Link-Prefix AVP	5
4.4. MIP6-Feature-Vector AVP	5
5. Example Signaling Flows for Localized Routing Service Authorization	6
6. Security Considerations	9
7. IANA Considerations	10
8. Contributors	10
9. Acknowledgements	10
10. References	10
10.1. Normative References	10
10.2. Informative References	11
Authors' Addresses	11

1. Introduction

Proxy Mobile IPv6 (PMIPv6) [RFC5213] allows the Mobility Access Gateway (MAG) to optimize media delivery by locally routing packets from a Mobile Node to a Correspondent Node that is locally attached to an access link connected to the same Mobile Access Gateway, avoiding tunneling them to the Mobile Node's Local Mobility Anchor (LMA). This is referred to as "local routing" in RFC 5213. However, this mechanism is not applicable to the typical scenarios in which the MN and CN are connected to different MAGs and are registered to the same LMA or different LMAs. [RFC6279] takes those typical scenarios into account and defines the problem statement for PMIPv6 localized routing. [I-D.ietf-netext-pmip-lr] specifies the PMIPv6 localized routing protocol based on the scenarios A11, A12, and A21 [RFC6279], which is used to establish a localized routing path between two Mobile Access Gateways in a PMIPv6 domain.

However, there is no relevant work discussing how AAA-based mechanisms can be used to provide authorization to the Mobile Node's MAG or LMA for enabling localized routing between MAGs.

This document describes Diameter [I-D.ietf-dime-rfc3588bis] support for the authorization of PMIPv6 mobility entities in case of A11,A12,A21 during localized routing.

2. Terminology

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

3. Solution Overview

This document addresses how to provide authorization to the Mobile Node's MAG or LMA for enabling localized routing and resolve the destination MN's MAG by means of interaction between the LMA and the AAA server. Figure 1 shows the reference architecture for Localized Routing Service Authorization. This reference architecture assumes that

- o If MN and CN belong to different LMAs, MN and CN should share the same MAG (i.e., A12 described in [RFC6279]), e.g., MN1 and CN2 in Figure 1 are attached to the same MAG1 and belong to LMA1 and LMA2 respectively. Note that LMA1 and LMA2 in Figure 1 are in the same provider domain (as described in [RFC6279]).

- o If MN and CN are attached to the different MAGs, MN and CN should belong to the same LMA (i.e., A21 described in [RFC6279]), e.g., MN1 and CN3 in the Figure 1 are attached to the MAG1 and MAG3 respectively but belong to LMA1.
- o MN and CN may belong to the same LMA and are attached to the same MAG (i.e., A11 described in [RFC6279]), e.g., MN1 and CN1 in the Figure 1 are both attached to the MAG1 and belong to LMA1.
- o The MAG and LMA support Diameter client functionality.

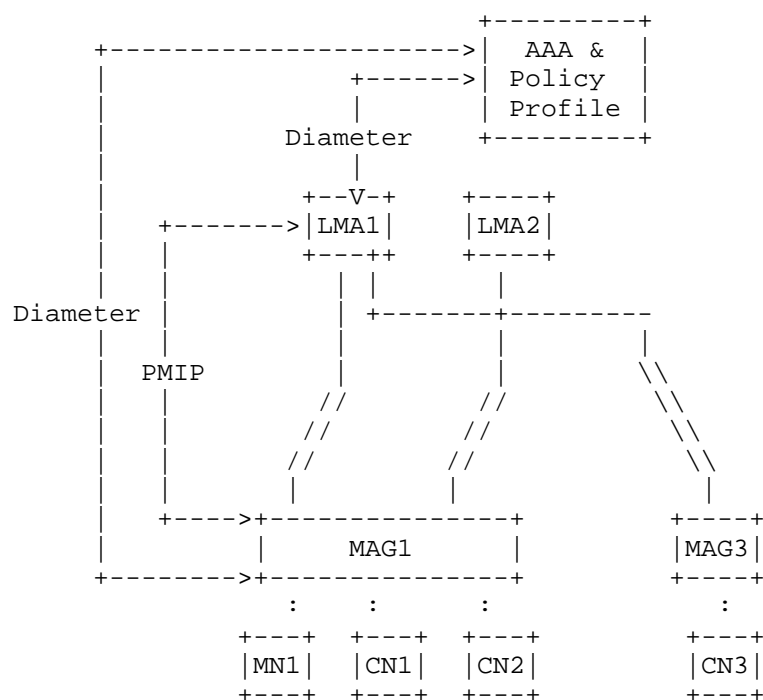


Figure 1: Localized Routing Service Authorization Reference Architecture

The interaction of the MAG and LMA with the AAA server according to the extension specified in this document is used to authorize the localized routing service.

4. Attribute Value Pair Used in this Document

This section describes Attribute Value Pairs (AVPs) defined by this

specification or re-used from existing specifications in a PMIPv6-specific way.

4.1. User-Name AVP

The User-Name AVP (AVP Code 1) is defined in [I-D.ietf-dime-rfc3588bis]. This AVP is used to carry the MN-Identifier (Mobile Node identifier) [RFC5213] in the AA-Request (AAR) message [I-D.ietf-dime-rfc4005bis].

4.2. PMIP6-IPv4-Home-Address AVP

The PMIP6-IPv4-Home-Address AVP (AVP Code 505) is defined in [RFC5779]. This AVP is used to carry the IPv4-MN-HoA (Mobile Node's IPv4 home address) [RFC5844] in the AA-Request (AAR) message [I-D.ietf-dime-rfc4005bis].

4.3. MIP6-Home-Link-Prefix AVP

The MIP6-Home-Link-Prefix AVP (AVP Code 125) is defined in [RFC5779]. This AVP is used to carry the MN-HNP (Mobile Node's home network prefix) in the AAR.

4.4. MIP6-Feature-Vector AVP

The MIP6-Feature-Vector AVP is defined in [RFC5447]. This document allocates a new capability flag bit according to the IANA rules in RFC 5447.

INTER_MAG_ROUTING_SUPPORTED (TBD)

Direct routing of IP packets between MNs anchored to different MAGs without involving any LMA is supported. This bit is used with MN-Identifier. When a MAG or LMA sets this bit in the MIP6-Feature-Vector and MN-Identifier corresponding to the Mobile Node is carried with this bit, it indicates to the home AAA server (HAAA) that the Mobile Node associated with this LMA is allowed to use localized routing. If this bit is cleared and MN-Identifier corresponding to the Mobile Node is carried with this bit, it indicates to the home AAA server (HAAA) that the Mobile Node associated with this LMA is not allowed to use localized routing. When a MAG or LMA sets this bit in the MIP6-Feature-Vector and MN-Identifiers corresponding to the Mobile Node and Correspondent Node are both carried with this bit, it indicates to the HAAA that localized routing of IP packets between Mobile Node and Correspondent Node anchored to different MAGs is supported. If this bit is cleared and MN-Identifiers corresponding to the Mobile Node and Correspondent Node are both carried with this bit

to HAAA, it indicates to the HAAA that localized routing of IP packets between Mobile Node and Correspondent Node anchored to different MAGs is not supported. If this bit is cleared in the returned MIP6-Feature-Vector AVP, the HAAA does not authorize direct routing of packets between MNs anchored to the different MAG. The MAG and LMA MUST support this policy feature on a per-MN and per-subscription basis.

5. Example Signaling Flows for Localized Routing Service Authorization

Localized Routing Service Authorization can happen during the network access authentication procedure [RFC5779] before localized routing is initialized. In this case, the preauthorized pairs of LMA/prefix sets can be downloaded to Proxy Mobile IPv6 entities during the RFC 5779 procedure. Localized routing can be initiated once the destination of a received packet matches one or more of the prefixes received during the RFC 5779 procedure.

Figure 2 shows an example scenario in which MAG1 acts as a Diameter client, processing the data packet from MN1 to MN2 and requesting authorization of localized routing (i.e., MAG-Initiated LR authorization). In this example scenario, MN1 and MN2 are attached to the same MAG and anchored to the different LMAs (i.e., A12 described in [RFC6279]). In this case, MAG1 knows that MN2 belongs to a different LMA (which can be determined by looking up the binding cache entries corresponding to MN1 and MN2 and comparing the addresses of LMA1 and LMA2). In order to setup a localized routing path with MAG2, MAG1 acts as Diameter client and sends an AAR message to the Diameter server. The message contains an instance of the MIP6-Feature-Vector (MFV) AVP ([RFC5447], Section 4.2.5) with the LOCAL_MAG_ROUTING_SUPPORTED bit ([RFC5779], Section 5.5) set, two instances of the User-Name AVP ([I-D.ietf-dime-rfc3588bis], Section 8.14) containing MN1-Identifier and MN2-Identifier. In addition, the message may contain either an instance of the MIP6-Home-Link-Prefix AVP ([RFC5779], Section 5.3) or an instance of the PMIP6-IPv4-Home-Address AVP ([RFC5779], Section 5.2) containing the IP address/ HNP of MN1.

The Diameter server authorizes localized routing service by checking if MN1 and MN2 are allowed to use localized routing. If so, the Diameter server responds with an AAA message encapsulating an instance of the MIP6-Feature-Vector (MFV) AVP ([RFC5447], Section 4.2.5) with the LOCAL_MAG_ROUTING_SUPPORTED bit ([RFC5779], Section 5.5) set indicating direct routing of IP packets between MNs anchored to the same MAG is supported. MAG1 then knows the localized routing between MN1 and MN2 is allowed. Then MAG1 sends the Request messages respectively to LMA1 and LMA2. The

request message is the Localized Routing Initialization (LRI) message in Figure 2 and belongs to the Initial phase of the localized routing. LMA1 and LMA2 responds to MAG1 using the Localized Routing Acknowledge message (LRA in Figure 2) in accordance with [I-D.ietf-netext-pmip-lr].

In case of LRA_WAIT_TIME expiration [I-D.ietf-netext-pmip-lr],MAG1 should ask for authorization of localized routing again according to the procedure described above before LRI is retransmitted up to a maximum of LRI_RETRIES.

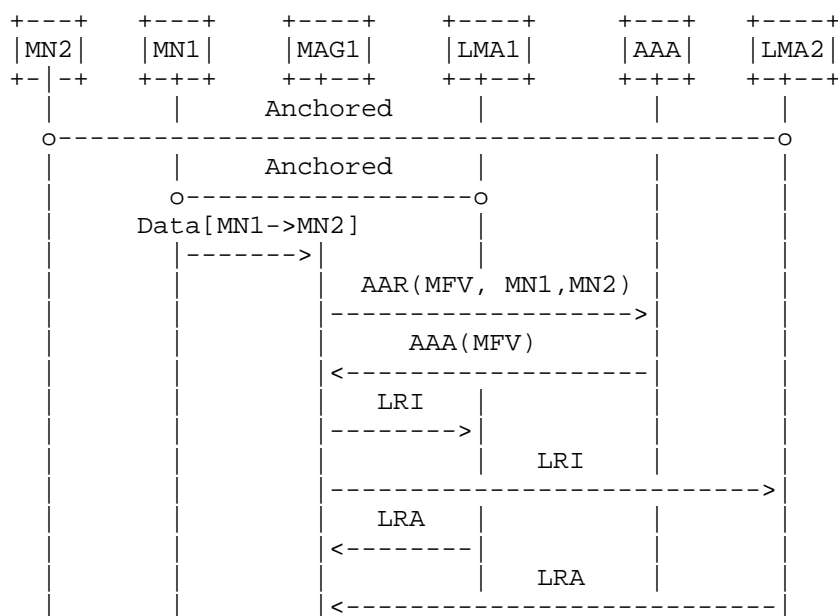


Figure 2: MAG-initiated Localized Routing Authorization in A12

Figure 3 shows the second example scenario, in which LMA1 acts as a Diameter client, processing the data packet from MN2 to MN1 and requesting the authorization of localized routing. In this scenario, MN1 and MN2 are attached to the different MAG and anchored to the same LMA (i.e., A21 described in [RFC6279]), LMA knows that MN1 and MN2 belong to the same LMA (which can be determined by looking up the binding cache entries corresponding to MN1 and MN2 and comparing the addresses of LMA corresponding to MN1 and LMA corresponding to MN2). In contrast with the signaling flow shown in Figure 2, it is LMA1 instead of MAG1 which initiates the setup of the localized routing path.

The Diameter client in LMA1 sends an AA-Request message to the Diameter server. The message contains an instance of the MIP6-Feature-Vector (MFV) AVP ([RFC5447], Section 4.2.5) with the INTER_MAG_ROUTING_SUPPORTED bit (Section 4.5) set indicating direct routing of IP packets between MNs anchored to different MAGs is supported and two instances of the User-Name AVP ([I-D.ietf-dime-rfc3588bis], Section 8.14) containing MN1-Identifier and MN2-Identifier. The Diameter server authorizes the localized routing service by checking if MN1 and MN2 are allowed to use localized routing. If so, the Diameter server responds with an AA-Answer message encapsulating an instance of the MIP6-Feature-Vector (MFV) AVP ([RFC5447], Section 4.2.5) with the INTER_MAG_ROUTING_SUPPORTED bit (Section 4.5) set indicating direct routing of IP packets between MNs anchored to different MAGs is supported. LMA1 then knows the localized routing is allowed. In success case, LMA1 responds to MAG1 in accordance with [I-D.ietf-netext-pmip-lr].

In case of LRA_WAIT_TIME expiration [I-D.ietf-netext-pmip-lr], LMA1 should ask for authorization of localized routing again according to the procedure described above before LRI is retransmitted up to a maximum of LRI_RETRIES.

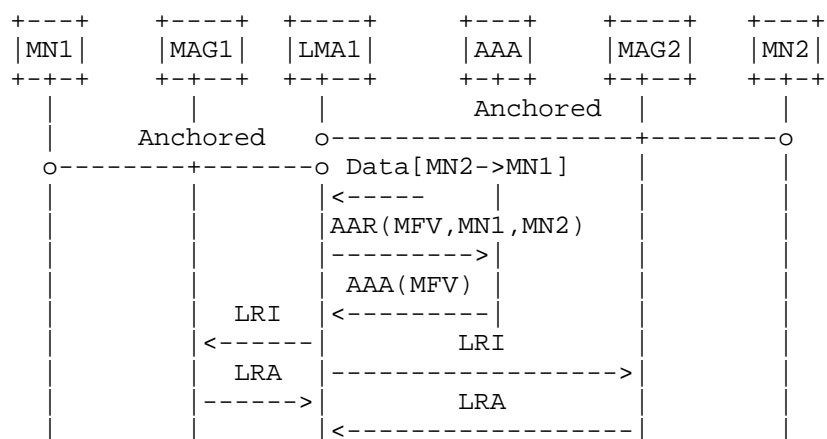


Figure 3: LMA-initiated Localized Routing Authorization in A21

Figure 4 shows another example scenario, in which LMA1 acts as a Diameter client, processing the data packet from MN2 to MN1 and requesting the authorization of localized routing. In this scenario, MN1 and MN2 are attached to the same MAG and anchored to the same LMA (i.e., A11 described in [RFC6279]), LMA knows that MN1 and MN2 belong to the same LMA (which can be determined by looking up the binding

cache entries corresponding to MN1 and MN2 and comparing the addresses of LMA corresponding to MN1 and LMA corresponding to MN2).

The Diameter client in LMA1 sends an AA-Request message to the Diameter server. The message contains an instance of the MIP6-Feature-Vector AVP ([RFC5447], Section 4.2.5) with the LOCAL_MAG_ROUTING_SUPPORTED bit set and two instances of the User-Name AVP ([I-D.ietf-dime-rfc3588bis], Section 8.14) containing MN1-Identifier and MN2-Identifier. The Diameter server authorizes the localized routing service by checking if MN1 and MN2 are allowed to use localized routing. If so, the Diameter server responds with an AA-Answer message encapsulating an instance of the MIP6-Feature-Vector (MFV) AVP ([RFC5447], Section 4.2.5) with the LOCAL_MAG_ROUTING_SUPPORTED bit ([RFC5779], Section 5.5) set indicating direct routing of IP packets between MNs anchored to the same MAG is supported. LMA1 then knows the localized routing is allowed and responds to MAG1 for localized routing in accordance with [I-D.ietf-netext-pmip-lr].

In case of LRA_WAIT_TIME expiration [I-D.ietf-netext-pmip-lr], LMA1 should ask for authorization of localized routing again according to the procedure described above before LRI is retransmitted up to a maximum of LRI_RETRIES.

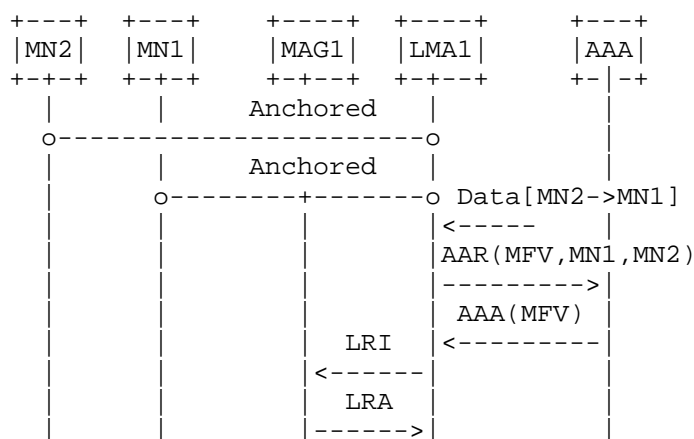


Figure 4: LMA-initiated Localized Routing Authorization in All

6. Security Considerations

The security considerations for the Diameter NASREQ [I-D.ietf-dime-rfc4005bis] and Diameter Proxy Mobile IPv6 [RFC5779] applications are also applicable to this document.

The service authorization solicited by the MAG or the LMA relies upon the existing trust relationship between the MAG/LMA and the AAA server.

An authorised MAG could in principle track the movement of any participating CNs at the level of the MAG to which they are anchored. If such a MAG were compromised, or under the control of a bad-actor, then such tracking could represent a privacy breach for the set of tracked CNs. In such a case, the traffic pattern from the compromised MAG might be notable so monitoring for e.g. excessive queries from MAGs might be worthwhile.

7. IANA Considerations

This specification defines a new value in the Mobility Capability registry [RFC5447] for use with the MIP6-Feature-Vector AVP: INTER_MAG_ROUTING_SUPPORTED (see Section 4.4).

8. Contributors

Paulo Loureiro, Jinwei Xia and Yungui Wang all contributed to early versions of this document.

9. Acknowledgements

The authors would like to thank Marco Liebsch, Carlos Jesus Bernardos Cano, Dan Romascanu, Elwyn Davies, Basavaraj Patil, Ralph Droms, Stephen Farrel, Robert Sparks, Benoit Claise and Abhay Roy for their valuable comments and suggestions on this document.

10. References

10.1. Normative References

[I-D.ietf-dime-rfc3588bis]

Fajardo, V., Arkko, J., Loughney, J., and G. Zorn,
"Diameter Base Protocol", draft-ietf-dime-rfc3588bis-34
(work in progress), June 2012.

[I-D.ietf-dime-rfc4005bis]

Zorn, G., "Diameter Network Access Server Application",
draft-ietf-dime-rfc4005bis-11 (work in progress),
July 2012.

- [I-D.ietf-netext-pmip-lr]
Krishnan, S., Koodli, R., Loureiro, P., Wu, Q., and A. Dutta, "Localized Routing for Proxy Mobile IPv6", draft-ietf-netext-pmip-lr-10 (work in progress), May 2012.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC5213] Gundavelli, S., Leung, K., Devarapalli, V., Chowdhury, K., and B. Patil, "Proxy Mobile IPv6", RFC 5213, August 2008.
- [RFC5447] Korhonen, J., Bournelle, J., Tschofenig, H., Perkins, C., and K. Chowdhury, "Diameter Mobile IPv6: Support for Network Access Server to Diameter Server Interaction", RFC 5447, February 2009.
- [RFC5779] Korhonen, J., Bournelle, J., Chowdhury, K., Muhanna, A., and U. Meyer, "Diameter Proxy Mobile IPv6: Mobile Access Gateway and Local Mobility Anchor Interaction with Diameter Server", RFC 5779, February 2010.
- [RFC5844] Wakikawa, R. and S. Gundavelli, "IPv4 Support for Proxy Mobile IPv6", RFC 5844, May 2010.

10.2. Informative References

- [RFC6279] Liebsch, M., Jeong, S., and Q. Wu, "Proxy Mobile IPv6 (PMIPv6) Localized Routing Problem Statement", RFC 6279, June 2011.

Authors' Addresses

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

Phone: +66 (0) 87-040-4617
Email: glenzorn@gmail.com

Qin Wu
Huawei Technologies Co., Ltd.
101 Software Avenue, Yuhua District
Nanjing, Jiangsu 21001
China

Phone: +86-25-84565892
Email: sunseawq@huawei.com

Jouni Korhonen
Nokia Siemens Networks
Linnoitustie 6
Espoo FI-02600,
Finland

Email: jouni.nospam@gmail.com

Internet Engineering Task Force
Internet-Draft
Updates: 6733 (if approved)
Intended status: Standards Track
Expires: April 04, 2014

T. Tsou
Huawei Technologies (USA)
R. Hao
Comcast Cable
T. Taylor, Ed.
Huawei Technologies
October 01, 2013

Realm-Based Redirection In Diameter
draft-ietf-dime-realm-based-redirect-13

Abstract

The Diameter protocol includes a capability for message redirection, controlled by an application-independent "redirect agent". In some circumstances, an operator may wish to redirect messages to an alternate domain without specifying individual hosts. This document specifies an application-specific mechanism by which a Diameter server or proxy (node) can perform such a redirection when S-NAPTR is not used for dynamic peer discovery. A node performing this new function is referred to as a "Realm-based Redirect Server".

This memo updates Sections 6.13 and 6.14 of RFC6733 with respect to the usage of the Redirect-Host-Usage and Redirect-Max-Cache-Time AVPs.

Status of This Memo

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

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

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

This Internet-Draft will expire on April 04, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
1.1. Terminology	3
2. Support of Realm-Based Redirection	3
Applications	3
3. Realm-Based Redirection	4
3.1. Configuration of the Realm-based Redirect Server	5
3.2. Behaviour of Diameter Nodes	5
3.2.1. Behaviour at the Realm-based Redirect Server	5
3.2.2. Proxy Behaviour	6
3.2.3. Client Behaviour	6
3.3. The Redirect-Realm AVP	7
3.4. DIAMETER_REALM_REDIRECT_INDICATION Protocol Error Code	7
4. Security Considerations	7
5. IANA Considerations	8
6. Acknowledgements	8
7. References	9
7.1. Normative References	9
7.2. Informative References	9
Authors' Addresses	9

1. Introduction

The Diameter base protocol [RFC6733] specifies a basic redirection service provided by redirect agent. The redirect indication returned by the redirect agent is described in Section 6.1.8 and Sections 6.12-6.14 of [RFC6733], and provides to the message sender one or more individual hosts as destination of the redirected message.

However, consider the case where an operator has offered a specific service but no longer wishes to do so. The operator has arranged for an alternative domain to provide the service. To aid in the transition to the new arrangement, the original operator maintains a redirect server to indicate to the message sender the alternative domain to which redirect the request. However, the original operator should be relieved from configuring in the redirect server a list of hosts to contact in the alternative operator's domain, and should

simply be able to provide redirect indications to the domain as a whole.

1.1. Terminology

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

Within this specification, the term "realm-based redirection" is used to refer to a mode of operation where a realm rather than an individual host is returned as redirect indication.

The term "Realm-based Redirect Server" denotes the Diameter node (Diameter server or proxy) that returns the realm-based redirection. The behaviour of the Realm-based Redirect Server itself is a slight modification of the behaviour of a basic redirect agent as described in Section 6.1.8 of [RFC6733].

This document uses a number of terms consistently with their usage in [RFC6733]: "Diameter client", "Diameter node", "Diameter peer", "Diameter server", "proxy", "realm" or "domain", "redirect agent", and "session" as defined in Section 1.2, and "application" as defined implicitly by Sections 1.3.4, 2.3, and 2.4.

2. Support of Realm-Based Redirection Within Applications

The DNS-based dynamic peer discovery mechanism defined in the Diameter base protocol [RFC6733] provides a simple mechanism for realm-based redirection, using the S-NAPTR DDDS application [RFC3958]. When S-NAPTR is used for peer discovery, redirection of Diameter requests from the original realm to a new realm may be performed by updating the existing NAPTR resource records for the original realm as follows: the NAPTR RR for the desired application(s) and supported application protocol(s) provided by the new realm will have an empty FLAG field and the REPLACEMENT field will contain the new realm to use for the next DNS lookup. The new realm can be arbitrary; the restriction in [RFC6733] that the NAPTR replacement field match the domain of the original query does not apply for realm-based redirect purposes.

However, the use of DNS-based dynamic peer discovery is optional for Diameter implementations. For deployments which do not make use of S-NAPTR peer discovery, support of realm-based redirection needs to be specified as part of functionality supported by a Diameter application. In this way, support of the considered Diameter application (discovered during capabilities exchange procedures as defined in Diameter base protocol [RFC6733]) indicates implicit

support of the realm-based redirection mechanism. A new application specification can incorporate the mechanism specified here by making it mandatory to implement for the application, and referencing this specification normatively.

The result of making realm-based redirection an application-specific behaviour is that it cannot be performed by a redirect agent as defined in [RFC6733], but **MUST** be performed instead by an application-aware Diameter node (Diameter server or proxy) (hereafter called a "Realm-based Redirect Server").

An application can specify that realm-based redirection operates only on complete sessions beginning with the initial message, or on every message within the application, even if earlier messages of the same session were not redirected. This distinction matters only when realm-based redirection is first initiated. In the former case, existing sessions will not be disrupted by the deployment of realm-based redirection. In the latter case, existing sessions will be disrupted if they are stateful.

3. Realm-Based Redirection

This section specifies an extension of the Diameter base protocol [RFC6733] to achieve realm-based redirection. The elements of this solution are:

- o a new result code, `DIAMETER_REALM_REDIRECT_INDICATION` (3xxx TBD1);
- o a new attribute-value pair (AVP), `Redirect-Realm` (code TBD2); and
- o associated behaviour at Diameter nodes implementing this specification.

This behaviour includes the optional use of the `Redirect-Host-Usage` and `Redirect-Max-Cache-Time` AVPs. In this document, these AVPs apply to the peer discovered by a node acting on the redirect server's response, an extension to their normal usage as described in Sections 6.13 and 6.14 of [RFC6733].

Section 3.2.2 and Section 3.2.3 describe how a proxy or client may update its routing table for the application and initial realm, as a result of selecting a peer in the new realm after realm-based redirection. Note that as a result, the proxy or client will automatically route subsequent requests for that application to the new realm (with the possible exception of requests within sessions already established with the initial realm) until the cached routing entry expires. This should be borne in mind if the rerouting is intended to be temporary.

3.1. Configuration of the Realm-based Redirect Server

A Diameter node (Diameter server or proxy) acting as Realm-based Redirect Server MUST be configured as follows to execute realm-based redirection:

- o configured with an application that incorporates realm-based redirection;
- o the Local Action field of the routing table described in Section 2.7 of [RFC6733] is set to LOCAL;
- o an application-specific field is set to indicate that the required local action is to perform realm-based redirection;
- o an associated application-specific field is configured with the identities of one or more realms to which the request should be redirected.

3.2. Behaviour of Diameter Nodes

3.2.1. Behaviour at the Realm-based Redirect Server

As mentioned in Section 2, an application can specify that realm-based redirection operates only on complete sessions beginning with the initial message (i.e., to prevent disruption of established sessions), or on every message within the application, even if earlier messages of the same session were not redirected.

If a Realm-based Redirect Server configured as described in Section 3.1 receives a request to which realm-based redirection applies, the Realm-based Redirect Server MUST reply with an answer message with the 'E' bit set, while maintaining the Hop-by-Hop Identifier in the header. The Realm-based Redirect Server MUST include the Result-Code AVP set to DIAMETER_REALM_REDIRECT_INDICATION. The Realm-based Redirect Server MUST also include the alternate realm identifier(s) with which it has been configured, each in a separate Redirect-Realm AVP instance.

The Realm-based Redirect Server MAY include a copy of the Redirect-Host-Usage AVP, which SHOULD be set to REALM_AND_APPLICATION. If this AVP is added, the Redirect-Max-Cache-Time AVP MUST also be included. Note that these AVPs apply to the peer discovered by a node acting on the Realm-based Redirect Server's response, as described in the next section. This is an extension of their normal usage as described by Sections 6.13 and 6.14 of [RFC6733].

Realm-based redirection MAY be applied even if a Destination-Host AVP is present in the request, depending on the operator-based policy.

3.2.2. Proxy Behaviour

A proxy conforming to this specification that receives an answer message with the Result-Code AVP set to `DIAMETER_REALM_REDIRECT_INDICATION` MUST attempt to reroute the original request to a server in a realm identified by a Redirect-Realm AVP instance in the answer message, and if it fails MUST forward the indication toward the client. To reroute the request, it MUST take the following actions:

1. Select a specific realm from amongst those identified in instances of the Redirect-Realm AVP in the answer message.
2. If successful, locate and establish a route to a peer in the realm given by the Redirect-Realm AVP, using normal discovery procedures as described in Section 5.2 of [RFC6733].
3. If again successful:
 - a. update its cache of routing entries for the realm and application to which the original request was directed, taking into account the Redirect-Host-Usage and Redirect-Max-Cache-Time AVPs, if present in the answer.
 - b. Remove the Destination-Host (if present) and Destination-Realm AVPs from the original request and add a new Destination-Realm AVP containing the realm selected in the initial step.
 - c. Forward the modified request.
4. If either of the preceding steps 2-3 fail and additional realms have been identified in the original answer, select another instance of the Redirect-Realm AVP in that answer and repeat steps 2-3 for the realm that it identifies.

3.2.3. Client Behaviour

A client conforming to this specification MUST be prepared to receive either an answer message containing a Result-Code AVP set to `DIAMETER_REALM_REDIRECT_INDICATION`, or, as the result of proxy action, some other result from a realm differing from the one to which it sent the original request. In the case where it receives `DIAMETER_REALM_REDIRECT_INDICATION`, the client SHOULD follow the same

steps prescribed in the previous section for a proxy, in order both to update its routing table and to obtain service for the original request.

3.3. The Redirect-Realm AVP

The Redirect-Realm AVP (code TBD2) is of type DiameterIdentity. It specifies a realm to which a node receiving a redirect indication containing the result code value `DIAMETER_REALM_REDIRECT_INDICATION` and the Redirect-Realm AVP SHOULD route the original request.

3.4. `DIAMETER_REALM_REDIRECT_INDICATION` Protocol Error Code

The `DIAMETER_REALM_REDIRECT_INDICATION` (3xxx TBD1) Protocol error code indicates that a server has determined that the request within an application supporting realm-based redirection could not be satisfied locally and the initiator of the request SHOULD direct the request directly to a peer within a realm that has been identified in the response. When set, the Redirect-Realm AVP MUST be present.

4. Security Considerations

The general recommendations given in the section 13 of the Diameter base protocol [RFC6733] apply. Specific security recommendations related to the realm-based redirection defined in this document are described below.

Realm-based redirection implies a change of the business relationships between organizations. Before redirecting a request towards a realm different from the initial realm, the client or proxy MUST ensure that the authorization checks have been performed at each connection along the path toward the realm identified in the realm-based redirect indication. Details on Diameter authorization path set-up are given in section 2.9 of [RFC6733]. Section 13 of [RFC6733] provides recommendations on how to authenticate and secure each peer-to-peer connection (using on TLS, DTLS or IPsec) along the way, thus permitting the necessary hop-by-hop authorization checks.

Although it is assumed that the administrative domains are secure, a compromised Diameter node acting as Realm-Based Redirect Server would be able to redirect a large number of Diameter requests towards a victim domain which would then be flooded with undesired Diameter requests. Such an attack is nevertheless discouraged by the use of secure Diameter peer-to-peer connections and authorization checks, since these would enable a potential victim domain to discover from where an attack is coming. That in itself, however, does not prevent such a DoS attack.

Because realm-based redirection defined in this document implies that the Destination-Realm AVP in a client-initiated request can be changed by a Diameter proxy in the path between the client and the server, any cryptographic algorithm that would use the Destination-Realm AVP as input to the calculation performed by the client and the server would be broken by this form of redirection. Application specifications that would rely on such cryptographic algorithm SHOULD NOT incorporate this realm-based redirection.

5. IANA Considerations

This specification adds a new AVP code [TBD2] Redirect-Realm in the AVP Code registry under Authentication, Authorization, and Accounting (AAA) Parameters.

This specification allocates a new Result-Code value `DIAMETER_REALM_REDIRECT_INDICATION` (3xxx TBD1) in the Result-Code AVP Values (code 268) - Protocol Errors registry under Authentication, Authorization, and Accounting (AAA) Parameters.

6. Acknowledgements

Glen Zorn, Sebastien Decugis, Wolfgang Steigerwald, Mark Jones, Victor Fajardo, Jouni Korhonen, Avi Lior, and Lionel Morand contributed comments that helped to shape this document. As shepherd, Lionel contributed a second set of comments that added polish to the document before it was submitted to the IESG. Benoit Claise picked up additional points which were quickly resolved with Lionel's help. During IETF Last Call Review, Enrico Marocco picked up some important editorial corrections. Stefan Winter contributed text on the use of S-NAPTR as an alternative method of realm-based redirection already specified in [RFC6733]. Derek Atkins performed a review on behalf of the Security Directorate. Lionel noted one more correction.

Finally, this document benefited from comments and discussion raised by IESG members Stewart Bryant, Stephen Farrell, Barry Leiba, Pete Resnick, Jaari Arkko, and Sean Turner during IESG review.

The authors are very grateful to Lionel Morand for his active role as document shepherd. At each stage, he worked to summarize and resolve comments, making the editor's role easy. Thank you.

7. References

7.1. Normative References

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.

7.2. Informative References

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

Authors' Addresses

Tina Tsou
Huawei Technologies (USA)
2330 Central Expressway
Santa Clara, CA 95050
USA

Phone: +1 408 330 4424
Email: Tina.Tsou.Zouting@huawei.com
URI: <http://tinatsou.weebly.com/contact.html>

Ruibing Hao
Comcast Cable
One Comcast Center
Philadelphia, PA 19103
USA

Phone: +1 215 286 3991(O)
Email: Ruibing_Hao@cable.comcast.com

Tom Taylor (editor)
Huawei Technologies
Ottawa
Canada

Email: tom.taylor.stds@gmail.com

Network Working Group
Internet-Draft
Obsoletes: 4005 (if approved)
Intended status: Standards Track
Expires: June 1, 2014

G. Zorn, Ed.
Network Zen
November 28, 2013

Diameter Network Access Server Application
draft-ietf-dime-rfc4005bis-14

Abstract

This document describes the Diameter protocol application used for Authentication, Authorization, and Accounting (AAA) services in the Network Access Server (NAS) environment; it obsoletes RFC 4005. When combined with the Diameter Base protocol, Transport Profile, and Extensible Authentication Protocol specifications, this application specification satisfies typical network access services requirements.

Status of This Memo

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

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

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

This Internet-Draft will expire on June 1, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1.	Introduction	4
1.1.	Changes from RFC 4005	5
1.2.	Terminology	6
1.3.	Requirements Language	7
1.4.	Advertising Application Support	8
1.5.	Application Identification	8
1.6.	Accounting Model	8
2.	NAS Calls, Ports, and Sessions	8
2.1.	Diameter Session Establishment	8
2.2.	Diameter Session Reauthentication or Reauthorization	9
2.3.	Diameter Session Termination	10
3.	Diameter NAS Application Messages	10
3.1.	AA-Request (AAR) Command	11
3.2.	AA-Answer (AAA) Command	12
3.3.	Re-Auth-Request (RAR) Command	14
3.4.	Re-Auth-Answer (RAA) Command	15
3.5.	Session-Termination-Request (STR) Command	16
3.6.	Session-Termination-Answer (STA) Command	17
3.7.	Abort-Session-Request (ASR) Command	17
3.8.	Abort-Session-Answer (ASA) Command	18
3.9.	Accounting-Request (ACR) Command	19
3.10.	Accounting-Answer (ACA) Command	21
4.	Diameter NAS Application AVPs	22
4.1.	Derived AVP Data Formats	22
4.1.1.	QoSFilterRule	22
4.2.	NAS Session AVPs	23
4.2.1.	Call and Session Information	24
4.2.2.	NAS-Port AVP	24
4.2.3.	NAS-Port-Id AVP	25
4.2.4.	NAS-Port-Type AVP	25
4.2.5.	Called-Station-Id AVP	25
4.2.6.	Calling-Station-Id AVP	25
4.2.7.	Connect-Info AVP	26
4.2.8.	Originating-Line-Info AVP	26
4.2.9.	Reply-Message AVP	27
4.3.	NAS Authentication AVPs	27
4.3.1.	User-Password AVP	28
4.3.2.	Password-Retry AVP	28
4.3.3.	Prompt AVP	28
4.3.4.	CHAP-Auth AVP	29
4.3.5.	CHAP-Algorithm AVP	29
4.3.6.	CHAP-Ident AVP	29
4.3.7.	CHAP-Response AVP	29

4.3.8.	CHAP-Challenge AVP	29
4.3.9.	ARAP-Password AVP	30
4.3.10.	ARAP-Challenge-Response AVP	30
4.3.11.	ARAP-Security AVP	30
4.3.12.	ARAP-Security-Data AVP	30
4.4.	NAS Authorization AVPs	30
4.4.1.	Service-Type AVP	32
4.4.2.	Callback-Number AVP	33
4.4.3.	Callback-Id AVP	33
4.4.4.	Idle-Timeout AVP	33
4.4.5.	Port-Limit AVP	33
4.4.6.	NAS-Filter-Rule AVP	33
4.4.7.	Filter-Id AVP	34
4.4.8.	Configuration-Token AVP	34
4.4.9.	QoS-Filter-Rule AVP	34
4.4.10.	Framed Access Authorization AVPs	35
4.4.10.1.	Framed-Protocol AVP	35
4.4.10.2.	Framed-Routing AVP	35
4.4.10.3.	Framed-MTU AVP	36
4.4.10.4.	Framed-Compression AVP	36
4.4.10.5.	IP Access Authorization AVPs	36
4.4.10.5.1.	Framed-IP-Address AVP	36
4.4.10.5.2.	Framed-IP-Netmask AVP	36
4.4.10.5.3.	Framed-Route AVP	37
4.4.10.5.4.	Framed-Pool AVP	37
4.4.10.5.5.	Framed-Interface-Id AVP	37
4.4.10.5.6.	Framed-IPv6-Prefix AVP	38
4.4.10.5.7.	Framed-IPv6-Route AVP	38
4.4.10.5.8.	Framed-IPv6-Pool AVP	38
4.4.10.6.	IPX Access AVPs	38
4.4.10.6.1.	Framed-IPX-Network AVP	38
4.4.10.7.	AppleTalk Network Access AVPs	39
4.4.10.7.1.	Framed-AppleTalk-Link AVP	39
4.4.10.7.2.	Framed-AppleTalk-Network AVP	39
4.4.10.7.3.	Framed-AppleTalk-Zone AVP	39
4.4.10.8.	AppleTalk Remote Access AVPs	40
4.4.10.8.1.	ARAP-Features AVP	40
4.4.10.8.2.	ARAP-Zone-Access AVP	40
4.4.11.	Non-Framed Access Authorization AVPs	40
4.4.11.1.	Login-IP-Host AVP	40
4.4.11.2.	Login-IPv6-Host AVP	41
4.4.11.3.	Login-Service AVP	41
4.4.11.4.	TCP Services	41
4.4.11.4.1.	Login-TCP-Port AVP	41
4.4.11.5.	LAT Services	41
4.4.11.5.1.	Login-LAT-Service AVP	41
4.4.11.5.2.	Login-LAT-Node AVP	42
4.4.11.5.3.	Login-LAT-Group AVP	43

4.4.11.5.4. Login-LAT-Port AVP	43
4.5. NAS Tunneling AVPs	43
4.5.1. Tunneling AVP	44
4.5.2. Tunnel-Type AVP	44
4.5.3. Tunnel-Medium-Type AVP	45
4.5.4. Tunnel-Client-Endpoint AVP	45
4.5.5. Tunnel-Server-Endpoint AVP	46
4.5.6. Tunnel-Password AVP	47
4.5.7. Tunnel-Private-Group-Id AVP	47
4.5.8. Tunnel-Assignment-Id AVP	47
4.5.9. Tunnel-Preference AVP	48
4.5.10. Tunnel-Client-Auth-Id AVP	49
4.5.11. Tunnel-Server-Auth-Id AVP	49
4.6. NAS Accounting AVPs	49
4.6.1. Accounting-Input-Octets AVP	50
4.6.2. Accounting-Output-Octets AVP	51
4.6.3. Accounting-Input-Packets AVP	51
4.6.4. Accounting-Output-Packets AVP	51
4.6.5. Acct-Session-Time AVP	51
4.6.6. Acct-Authentic AVP	51
4.6.7. Accounting-Auth-Method AVP	52
4.6.8. Acct-Delay-Time AVP	52
4.6.9. Acct-Link-Count AVP	52
4.6.10. Acct-Tunnel-Connection AVP	53
4.6.11. Acct-Tunnel-Packets-Lost AVP	53
5. AVP Occurrence Tables	53
5.1. AA-Request/Answer AVP Table	54
5.2. Accounting AVP Tables	56
5.2.1. Framed Access Accounting AVP Table	56
5.2.2. Non-Framed Access Accounting AVP Table	58
6. Unicode Considerations	60
7. IANA Considerations	60
8. Security Considerations	61
8.1. Authentication Considerations	61
8.2. AVP Considerations	62
9. References	62
9.1. Normative References	62
9.2. Informative References	63
Appendix A. Acknowledgements	66
A.1. This Document	66
A.2. RFC 4005	66

1. Introduction

This document describes the Diameter protocol application used for AAA in the Network Access Server (NAS) environment. When combined with the Diameter Base protocol [RFC6733], Transport Profile [RFC3539], and EAP [RFC4072] specifications, this specification

satisfies the NAS-related requirements defined in Aboba, et al. [RFC2989] and Beadles & Mitton [RFC3169].

First, this document describes the operation of a Diameter NAS application. Then it defines the Diameter message Command-Codes. The following sections list the AVPs used in these messages, grouped by common usage. These are session identification, authentication, authorization, tunneling, and accounting. The authorization AVPs are further broken down by service type.

1.1. Changes from RFC 4005

This document obsoletes RFC 4005 and is not backward compatible with that document. An overview of some of the major changes is given below.

- o All of the material regarding RADIUS/Diameter protocol interactions has been removed; however, where AVPs are derived from RADIUS Attributes, the range and format of those Attribute values have been retained for ease of transition.
- o The Command Code Format (CCF) [RFC6733] for the Accounting-Request and Accounting-Answer messages has been changed to explicitly require the inclusion of the Acct-Application-Id AVP and exclude the Vendor-Specific-Application-Id AVP. Normally, this type of change would require the allocation of a new command code and consequently, a new application-id (See Section 1.3.3 of [RFC6733]). However, the presence of an instance of the Acct-Application-Id AVP was required in RFC 4005, as well:

The ACR message [BASE] is sent by the NAS to report its session information to a target server downstream.

Either of Acct-Application-Id or Vendor-Specific-Application-Id AVPs MUST be present. If the Vendor-Specific-Application-Id grouped AVP is present, it must have an Acct-Application-Id inside.

Thus, though the syntax of the commands has changed, the semantics have not (with the caveat that the Acct-Application-Id AVP can no longer be contained in the Vendor-Specific-Application-Id AVP).

- o The lists of RADIUS attribute values have been deleted in favor of references to the appropriate IANA registries.
- o The accounting model to be used is now specified (see Section 1.6).

There are many other miscellaneous fixes that have been introduced in this document that may not be considered significant but they are useful nonetheless. Examples are fixes to example IP addresses, addition of clarifying references, etc. All of the errata previously filed against RFC 4005 have been fixed. A comprehensive list of changes is not shown here for practical reasons.

1.2. Terminology

Section 1.2 of the Diameter base protocol specification [RFC6733] defines most of the terminology used in this document. Additionally, the following terms and acronyms are used in this application:

NAS (Network Access Server)

A device that provides an access service for a user to a network. The service may be a network connection or a value-added service such as terminal emulation [RFC2881].

PPP (Point-to-Point Protocol)

A multiprotocol serial datalink. PPP is the primary IP datalink used for dial-in NAS connection service [RFC1661].

CHAP (Challenge Handshake Authentication Protocol)

An authentication process used in PPP [RFC1994].

PAP (Password Authentication Protocol)

A deprecated PPP authentication process, but often used for backward compatibility [RFC1334].

SLIP (Serial Line Interface Protocol)

A serial datalink that only supports IP. A design prior to PPP.

ARAP (Appletalk Remote Access Protocol)

A serial datalink for accessing Appletalk networks [ARAP].

IPX (Internet Packet Exchange)

The network protocol used by NetWare networks [IPX].

L2TP (Layer Two Tunneling Protocol)

L2TP [RFC3931] provides a dynamic mechanism for tunneling Layer 2 "circuits" across a packet-oriented data network.

LAC (L2TP Access Concentrator)

An L2TP Control Connection Endpoint being used to cross-connect an L2TP session directly to a data link [RFC3931].

LAT (Local Area Transport)

A Digital Equipment Corp. LAN protocol for terminal services [LAT].

LCP (Link Control Protocol)

One of the three major components of PPP [RFC1661]. LCP is used to automatically agree upon encapsulation format options, handle varying limits on sizes of packets, detect a looped-back link and other common misconfiguration errors, and terminate the link. Other optional facilities provided are authentication of the identity of its peer on the link, and determination when a link is functioning properly and when it is failing.

PPTP (Point-to-Point Tunneling Protocol)

A protocol which allows PPP to be tunneled through an IP network [RFC2637].

VPN (Virtual Private Network)

In this document, this term is used to describe access services that use tunneling methods.

1.3. Requirements Language

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

The use of "MUST" and "MUST NOT" in the AVP Flag rules columns of AVP Tables in this document refers to AVP flags ([RFC6733], Section 4.1) that:

- o MUST be set to 1 in the AVP Header ("MUST" column) and
- o MUST NOT be set to 1 ("MUST NOT" column)

1.4. Advertising Application Support

Diameter nodes conforming to this specification MUST advertise support by including the value of one (1) in the Auth-Application-Id of the Capabilities-Exchange-Request (CER) message [RFC6733].

1.5. Application Identification

When used in this application, the Auth-Application-Id AVP MUST be set to the value one (1) in the following messages

- o AA-Request (Section 3.1)
- o Re-Auth-Request (Section 3.3)
- o Session-Termination-Request (Section 3.5)
- o Abort-Session-Request (Section 3.7)

1.6. Accounting Model

It is RECOMMENDED that the coupled accounting model (RFC 6733, Section 9.3) be used with this application; therefore, the value of the Acct-Application-Id AVP in the Accounting-Request (Section 3.10) and Accounting-Answer (Section 3.9) messages SHOULD be set to one (1).

2. NAS Calls, Ports, and Sessions

The arrival of a new call or service connection at a port of a Network Access Server (NAS) starts a Diameter NAS Application message exchange. Information about the call, the identity of the user, and the user's authentication information are packaged into a Diameter AA-Request (AAR) message and sent to a server.

The server processes the information and responds with a Diameter AA-Answer (AAA) message that contains authorization information for the NAS, or a failure code (Result-Code AVP). A value of DIAMETER_MULTI_ROUND_AUTH indicates an additional authentication exchange, and several AAR and AAA messages may be exchanged until the transaction completes.

2.1. Diameter Session Establishment

When the authentication or authorization exchange completes successfully, the NAS application SHOULD start a session context. If the Result-Code of DIAMETER_MULTI_ROUND_AUTH is returned, the exchange continues until a success or error is returned.

If accounting is active, the application MUST also send an Accounting message [RFC6733]. An Accounting-Record-Type of START_RECORD is sent for a new session. If a session fails to start, the EVENT_RECORD message is sent with the reason for the failure described.

Note that the return of an unsupportable Accounting-Realtime-Required value [RFC6733] would result in a failure to establish the session.

2.2. Diameter Session Reauthentication or Reauthorization

The Diameter Base protocol allows users to be periodically reauthenticated and/or reauthorized. In such instances, the Session-Id AVP in the AAR message MUST be the same as the one present in the original authentication/authorization message.

A Diameter server informs the NAS of the maximum time allowed before reauthentication or reauthorization via the Authorization-Lifetime AVP [RFC6733]. A NAS MAY reauthenticate and/or reauthorize before the end, but A NAS MUST reauthenticate and/or reauthorize at the end of the period provided by the Authorization-Lifetime AVP. The failure of a reauthentication exchange will terminate the service.

Furthermore, it is possible for Diameter servers to issue an unsolicited reauthentication and/or reauthorization request (e.g., Re-Auth-Request (RAR) message [RFC6733]) to the NAS. Upon receipt of such a message, the NAS MUST respond to the request with a Re-Auth-Answer (RAA) message [RFC6733].

If the RAR properly identifies an active session, the NAS will initiate a new local reauthentication or authorization sequence as indicated by the Re-Auth-Request-Type value. This will cause the NAS to send a new AAR message using the existing Session-Id. The server will respond with an AAA message to specify the new service parameters.

If accounting is active, every change of authentication or authorization SHOULD generate an accounting message. If the NAS service is a continuation of the prior user context, then an Accounting-Record-Type of INTERIM_RECORD indicating the new session attributes and cumulative status would be appropriate. If a new user or a significant change in authorization is detected by the NAS, then the service may send two messages of the types STOP_RECORD and START_RECORD. Accounting may change the subsession identifiers (Acct-Session-ID, or Acct-Sub-Session-Id) to indicate such subsessions. A service may also use a different Session-Id value for accounting (see Section 9.6 of [RFC6733]).

However, the Diameter Session-ID AVP value used for the initial authorization exchange MUST be used to generate an STR message when the session context is terminated.

2.3. Diameter Session Termination

When a NAS receives an indication that a user's session is being disconnected by the client (e.g., an LCP Terminate-Request message [RFC1661] is received) or an administrative command, the NAS MUST issue a Session-Termination-Request (STR) [RFC6733] to its Diameter Server. This will ensure that any resources maintained on the servers are freed appropriately.

Furthermore, a NAS that receives an Abort-Session-Request (ASR) [RFC6733] MUST issue an Abort-Session-Answer (ASA) if the session identified is active and disconnect the PPP (or tunneling) session.

If accounting is active, an Accounting STOP_RECORD message [RFC6733] MUST be sent upon termination of the session context.

More information on Diameter Session Termination can be found in Sections 8.4 and 8.5 of [RFC6733].

3. Diameter NAS Application Messages

This section defines the Diameter message Command-Code [RFC6733] values that MUST be supported by all Diameter implementations conforming to this specification. The Command Codes are as follows:

Command Name	Abbrev.	Code	Reference
AA-Request	AAR	265	Section 3.1
AA-Answer	AAA	265	Section 3.2
Re-Auth-Request	RAR	258	Section 3.3
Re-Auth-Answer	RAA	258	Section 3.4
Session-Termination-Request	STR	275	Section 3.5
Session-Termination-Answer	STA	275	Section 3.6
Abort-Session-Request	ASR	274	Section 3.7
Abort-Session-Answer	ASA	274	Section 3.8
Accounting-Request	ACR	271	Section 3.9
Accounting-Answer	ACA	271	Section 3.10

Note that the message formats in the following sub-sections use the standard Diameter Command Code Format ([RFC6733], Section 3.2).

3.1. AA-Request (AAR) Command

The AA-Request (AAR), which is indicated by setting the Command-Code field to 265 and the 'R' bit in the Command Flags field, is used to request authentication and/or authorization for a given NAS user. The type of request is identified through the Auth-Request-Type AVP [RFC6733]. The recommended value for most situations is `AUTHORIZE_AUTHENTICATE`.

If Authentication is requested, the User-Name attribute SHOULD be present, as well as any additional authentication AVPs that would carry the password information. A request for authorization SHOULD only include the information from which the authorization will be performed, such as the User-Name, Called-Station-Id, or Calling-Station-Id AVPs. All requests SHOULD contain AVPs uniquely identifying the source of the call, such as Origin-Host and NAS-Port. Certain networks MAY use different AVPs for authorization purposes. A request for authorization will include some AVPs defined in Section 4.4.

It is possible for a single session to be authorized first and then for an authentication request to follow.

This AA-Request message MAY be the result of a multi-round authentication exchange, which occurs when the AA-Answer message is received with the Result-Code AVP set to `DIAMETER_MULTI_ROUND_AUTH`. A subsequent AAR message SHOULD be sent, with the User-Password AVP that includes the user's response to the prompt, and MUST include any State AVPs that were present in the AAA message.

Message Format

```
<AA-Request> ::= < Diameter Header: 265, REQ, PXY >
                  < Session-Id >
                  { Auth-Application-Id }
                  { Origin-Host }
                  { Origin-Realm }
                  { Destination-Realm }
                  { Auth-Request-Type }
                  [ Destination-Host ]
                  [ NAS-Identifier ]
                  [ NAS-IP-Address ]
                  [ NAS-IPv6-Address ]
                  [ NAS-Port ]
                  [ NAS-Port-Id ]
                  [ NAS-Port-Type ]
                  [ Origin-AAA-Protocol ]
                  [ Origin-State-Id ]
```

```
[ Port-Limit ]
[ User-Name ]
[ User-Password ]
[ Service-Type ]
[ State ]
[ Authorization-Lifetime ]
[ Auth-Grace-Period ]
[ Auth-Session-State ]
[ Callback-Number ]
[ Called-Station-Id ]
[ Calling-Station-Id ]
[ Originating-Line-Info ]
[ Connect-Info ]
[ CHAP-Auth ]
[ CHAP-Challenge ]
* [ Framed-Compression ]
[ Framed-Interface-Id ]
[ Framed-IP-Address ]
* [ Framed-IPv6-Prefix ]
[ Framed-IP-Netmask ]
[ Framed-MTU ]
[ Framed-Protocol ]
[ ARAP-Password ]
[ ARAP-Security ]
* [ ARAP-Security-Data ]
* [ Login-IP-Host ]
* [ Login-IPv6-Host ]
[ Login-LAT-Group ]
[ Login-LAT-Node ]
[ Login-LAT-Port ]
[ Login-LAT-Service ]
* [ Tunneling ]
* [ Proxy-Info ]
* [ Route-Record ]
* [ AVP ]
```

Figure 1

3.2. AA-Answer (AAA) Command

The AA-Answer (AAA) message is indicated by setting the Command-Code field to 265 and clearing the 'R' bit in the Command Flags field. It is sent in response to the AA-Request (AAR) message. If authorization was requested, a successful response will include the authorization AVPs appropriate for the service being provided, as defined in Section 4.4.

For authentication exchanges requiring more than a single round trip, the server MUST set the Result-Code AVP to DIAMETER_MULTI_ROUND_AUTH. An AAA message with this result code MAY include one Reply-Message or more and MAY include zero or one State AVPs.

If the Reply-Message AVP was present, the network access server SHOULD send the text to the user's client to display to the user, instructing the client to prompt the user for a response. For example, this can be achieved in PPP via PAP. If it is impossible to deliver the text prompt to the user, the Diameter NAS Application client MUST treat the AA-Answer (AAA) with the Reply-Message AVP as an error and deny access.

Message Format

```
<AA-Answer> ::= < Diameter Header: 265, PXY >
               < Session-Id >
               { Auth-Application-Id }
               { Auth-Request-Type }
               { Result-Code }
               { Origin-Host }
               { Origin-Realm }
               [ User-Name ]
               [ Service-Type ]
               * [ Class ]
               * [ Configuration-Token ]
               [ Acct-Interim-Interval ]
               [ Error-Message ]
               [ Error-Reporting-Host ]
               * [ Failed-AVP ]
               [ Idle-Timeout ]
               [ Authorization-Lifetime ]
               [ Auth-Grace-Period ]
               [ Auth-Session-State ]
               [ Re-Auth-Request-Type ]
               [ Multi-Round-Time-Out ]
               [ Session-Timeout ]
               [ State ]
               * [ Reply-Message ]
               [ Origin-AAA-Protocol ]
               [ Origin-State-Id ]
               * [ Filter-Id ]
               [ Password-Retry ]
               [ Port-Limit ]
               [ Prompt ]
               [ ARAP-Challenge-Response ]
               [ ARAP-Features ]
               [ ARAP-Security ]
```

```
* [ ARAP-Security-Data ]
  [ ARAP-Zone-Access ]
  [ Callback-Id ]
  [ Callback-Number ]
  [ Framed-Appletalk-Link ]
* [ Framed-Appletalk-Network ]
  [ Framed-Appletalk-Zone ]
* [ Framed-Compression ]
  [ Framed-Interface-Id ]
  [ Framed-IP-Address ]
* [ Framed-IPv6-Prefix ]
  [ Framed-IPv6-Pool ]
* [ Framed-IPv6-Route ]
  [ Framed-IP-Netmask ]
* [ Framed-Route ]
  [ Framed-Pool ]
  [ Framed-IPX-Network ]
  [ Framed-MTU ]
  [ Framed-Protocol ]
  [ Framed-Routing ]
* [ Login-IP-Host ]
* [ Login-IPv6-Host ]
  [ Login-LAT-Group ]
  [ Login-LAT-Node ]
  [ Login-LAT-Port ]
  [ Login-LAT-Service ]
  [ Login-Service ]
  [ Login-TCP-Port ]
* [ NAS-Filter-Rule ]
* [ QoS-Filter-Rule ]
* [ Tunneling ]
* [ Redirect-Host ]
  [ Redirect-Host-Usage ]
  [ Redirect-Max-Cache-Time ]
* [ Proxy-Info ]
* [ AVP ]
```

Figure 2

3.3. Re-Auth-Request (RAR) Command

A Diameter server can initiate re-authentication and/or re-authorization for a particular session by issuing a Re-Auth-Request (RAR) message [RFC6733].

For example, for pre-paid services, the Diameter server that originally authorized a session may need some confirmation that the user is still using the services.

If a NAS receives an RAR message with Session-Id equal to a currently active session and a Re-Auth-Type that includes authentication, it MUST initiate a re-authentication toward the user, if the service supports this particular feature.

Message Format

```

<RA-Request> ::= < Diameter Header: 258, REQ, PXY >
    < Session-Id >
    { Origin-Host }
    { Origin-Realm }
    { Destination-Realm }
    { Destination-Host }
    { Auth-Application-Id }
    { Re-Auth-Request-Type }
    [ User-Name ]
    [ Origin-AAA-Protocol ]
    [ Origin-State-Id ]
    [ NAS-Identifier ]
    [ NAS-IP-Address ]
    [ NAS-IPv6-Address ]
    [ NAS-Port ]
    [ NAS-Port-Id ]
    [ NAS-Port-Type ]
    [ Service-Type ]
    [ Framed-IP-Address ]
    [ Framed-IPv6-Prefix ]
    [ Framed-Interface-Id ]
    [ Called-Station-Id ]
    [ Calling-Station-Id ]
    [ Originating-Line-Info ]
    [ Acct-Session-Id ]
    [ Acct-Multi-Session-Id ]
    [ State ]
    * [ Class ]
    [ Reply-Message ]
    * [ Proxy-Info ]
    * [ Route-Record ]
    * [ AVP ]

```

Figure 3

3.4. Re-Auth-Answer (RAA) Command

The Re-Auth-Answer (RAA) message [RFC6733] is sent in response to the RAR. The Result-Code AVP MUST be present and indicates the disposition of the request.

A successful RAA transaction MUST be followed by an AAR message.

Message Format

```
<RA-Answer> ::= < Diameter Header: 258, PXY >
               < Session-Id >
               { Result-Code }
               { Origin-Host }
               { Origin-Realm }
               [ User-Name ]
               [ Origin-AAA-Protocol ]
               [ Origin-State-Id ]
               [ Error-Message ]
               [ Error-Reporting-Host ]
               * [ Failed-AVP ]
               * [ Redirected-Host ]
               [ Redirected-Host-Usage ]
               [ Redirected-Host-Cache-Time ]
               [ Service-Type ]
               * [ Configuration-Token ]
               [ Idle-Timeout ]
               [ Authorization-Lifetime ]
               [ Auth-Grace-Period ]
               [ Re-Auth-Request-Type ]
               [ State ]
               * [ Class ]
               * [ Reply-Message ]
               [ Prompt ]
               * [ Proxy-Info ]
               * [ AVP ]
```

Figure 4

3.5. Session-Termination-Request (STR) Command

The Session-Termination-Request (STR) message [RFC6733] is sent by the NAS to inform the Diameter Server that an authenticated and/or authorized session is being terminated.

Message Format

```
<ST-Request> ::= < Diameter Header: 275, REQ, PXY >
               < Session-Id >
               { Origin-Host }
               { Origin-Realm }
               { Destination-Realm }
               { Auth-Application-Id }
               { Termination-Cause }
```

```
[ User-Name ]
[ Destination-Host ]
* [ Class ]
[ Origin-AAA-Protocol ]
[ Origin-State-Id ]
* [ Proxy-Info ]
* [ Route-Record ]
* [ AVP ]
```

Figure 5

3.6. Session-Termination-Answer (STA) Command

The Session-Termination-Answer (STA) message [RFC6733] is sent by the Diameter Server to acknowledge the notification that the session has been terminated. The Result-Code AVP MUST be present and MAY contain an indication that an error occurred while the STR was being serviced.

Upon sending the STA, the Diameter Server MUST release all resources for the session indicated by the Session-Id AVP. Any intermediate server in the Proxy-Chain MAY also release any resources, if necessary.

Message Format

```
<ST-Answer> ::= < Diameter Header: 275, PXY >
               < Session-Id >
               { Result-Code }
               { Origin-Host }
               { Origin-Realm }
               [ User-Name ]
               * [ Class ]
               [ Error-Message ]
               [ Error-Reporting-Host ]
               * [ Failed-AVP ]
               [ Origin-AAA-Protocol ]
               [ Origin-State-Id ]
               * [ Redirect-Host ]
               [ Redirect-Host-Usase ]
               [ Redirect-Max-Cache-Time ]
               * [ Proxy-Info ]
               * [ AVP ]
```

Figure 6

3.7. Abort-Session-Request (ASR) Command

The Abort-Session-Request (ASR) message [RFC6733] can be sent by any Diameter server to the NAS providing session service to request that the session identified by the Session-Id be stopped.

Message Format

```

<AS-Request> ::= < Diameter Header: 274, REQ, PXY >
    < Session-Id >
    { Origin-Host }
    { Origin-Realm }
    { Destination-Realm }
    { Destination-Host }
    { Auth-Application-Id }
    [ User-Name ]
    [ Origin-AAA-Protocol ]
    [ Origin-State-Id ]
    [ NAS-Identifier ]
    [ NAS-IP-Address ]
    [ NAS-IPv6-Address ]
    [ NAS-Port ]
    [ NAS-Port-Id ]
    [ NAS-Port-Type ]
    [ Service-Type ]
    [ Framed-IP-Address ]
    [ Framed-IPv6-Prefix ]
    [ Framed-Interface-Id ]
    [ Called-Station-Id ]
    [ Calling-Station-Id ]
    [ Originating-Line-Info ]
    [ Acct-Session-Id ]
    [ Acct-Multi-Session-Id ]
    [ State ]
    * [ Class ]
    * [ Reply-Message ]
    * [ Proxy-Info ]
    * [ Route-Record ]
    * [ AVP ]

```

Figure 7

3.8. Abort-Session-Answer (ASA) Command

The ASA message [RFC6733] is sent in response to the ASR. The Result-Code AVP MUST be present and indicates the disposition of the request.

If the session identified by Session-Id in the ASR was successfully terminated, Result-Code is set to DIAMETER_SUCCESS. If the session

is not currently active, the Result-Code AVP is set to DIAMETER_UNKNOWN_SESSION_ID. If the access device does not stop the session for any other reason, the Result-Code AVP is set to DIAMETER_UNABLE_TO_COMPLY.

Message Format

```
<AS-Answer> ::= < Diameter Header: 274, PXY >
               < Session-Id >
               { Result-Code }
               { Origin-Host }
               { Origin-Realm }
               [ User-Name ]
               [ Origin-AAA-Protocol ]
               [ Origin-State-Id ]
               [ State ]
               [ Error-Message ]
               [ Error-Reporting-Host ]
               * [ Failed-AVP ]
               * [ Redirected-Host ]
               [ Redirected-Host-Usage ]
               [ Redirected-Max-Cache-Time ]
               * [ Proxy-Info ]
               * [ AVP ]
```

Figure 8

3.9. Accounting-Request (ACR) Command

The ACR message [RFC6733] is sent by the NAS to report its session information to a target server downstream.

The Acct-Application-Id AVP MUST be present.

The AVPs listed in the Base protocol specification [RFC6733] MUST be assumed to be present, as appropriate. NAS service-specific accounting AVPs SHOULD be present as described in Section 4.6 and the rest of this specification.

Message Format

```
<AC-Request> ::= < Diameter Header: 271, REQ, PXY >
               < Session-Id >
               { Origin-Host }
               { Origin-Realm }
               { Destination-Realm }
               { Accounting-Record-Type }
               { Accounting-Record-Number }
```

```
{ Acct-Application-Id }
[ User-Name ]
[ Accounting-Sub-Session-Id ]
[ Acct-Session-Id ]
[ Acct-Multi-Session-Id ]
[ Origin-AAA-Protocol ]
[ Origin-State-Id ]
[ Destination-Host ]
[ Event-Timestamp ]
[ Acct-Delay-Time ]
[ NAS-Identifier ]
[ NAS-IP-Address ]
[ NAS-IPv6-Address ]
[ NAS-Port ]
[ NAS-Port-Id ]
[ NAS-Port-Type ]
* [ Class ]
[ Service-Type ]
[ Termination-Cause ]
[ Accounting-Input-Octets ]
[ Accounting-Input-Packets ]
[ Accounting-Output-Octets ]
[ Accounting-Output-Packets ]
[ Acct-Authentic ]
[ Accounting-Auth-Method ]
[ Acct-Link-Count ]
[ Acct-Session-Time ]
[ Acct-Tunnel-Connection ]
[ Acct-Tunnel-Packets-Lost ]
[ Callback-Id ]
[ Callback-Number ]
[ Called-Station-Id ]
[ Calling-Station-Id ]
* [ Connection-Info ]
[ Originating-Line-Info ]
[ Authorization-Lifetime ]
[ Session-Timeout ]
[ Idle-Timeout ]
[ Port-Limit ]
[ Accounting-Realtime-Required ]
[ Acct-Interim-Interval ]
* [ Filter-Id ]
* [ NAS-Filter-Rule ]
* [ QoS-Filter-Rule ]
[ Framed-AppleTalk-Link ]
[ Framed-AppleTalk-Network ]
[ Framed-AppleTalk-Zone ]
[ Framed-Compression ]
```

```
[ Framed-Interface-Id ]
[ Framed-IP-Address ]
[ Framed-IP-Netmask ]
* [ Framed-IPv6-Prefix ]
[ Framed-IPv6-Pool ]
* [ Framed-IPv6-Route ]
[ Framed-IPX-Network ]
[ Framed-MTU ]
[ Framed-Pool ]
[ Framed-Protocol ]
* [ Framed-Route ]
[ Framed-Routing ]
* [ Login-IP-Host ]
* [ Login-IPv6-Host ]
[ Login-LAT-Group ]
[ Login-LAT-Node ]
[ Login-LAT-Port ]
[ Login-LAT-Service ]
[ Login-Service ]
[ Login-TCP-Port ]
* [ Tunneling ]
* [ Proxy-Info ]
* [ Route-Record ]
* [ AVP ]
```

Figure 9

3.10. Accounting-Answer (ACA) Command

The ACA message [RFC6733] is used to acknowledge an Accounting-Request command. The Accounting-Answer command contains the same Session-Id as the Request.

Only the target Diameter Server or home Diameter Server SHOULD respond with the Accounting-Answer command.

The Acct-Application-Id AVP MUST be present.

The AVPs listed in the Base protocol specification [RFC6733] MUST be assumed to be present, as appropriate. NAS service-specific accounting AVPs SHOULD be present as described in Section 4.6 and the rest of this specification.

Message Format

```
<AC-Answer> ::= < Diameter Header: 271, PXY >
               < Session-Id >
               { Result-Code }
```

```

{ Origin-Host }
{ Origin-Realm }
{ Accounting-Record-Type }
{ Accounting-Record-Number }
{ Acct-Application-Id }
[ User-Name ]
[ Accounting-Sub-Session-Id ]
[ Acct-Session-Id ]
[ Acct-Multi-Session-Id ]
[ Event-Timestamp ]
[ Error-Message ]
[ Error-Reporting-Host ]
* [ Failed-AVP ]
[ Origin-AAA-Protocol ]
[ Origin-State-Id ]
[ NAS-Identifier ]
[ NAS-IP-Address ]
[ NAS-IPv6-Address ]
[ NAS-Port ]
[ NAS-Port-Id ]
[ NAS-Port-Type ]
[ Service-Type ]
[ Termination-Cause ]
[ Accounting-Realtime-Required ]
[ Acct-Interim-Interval ]
* [ Class ]
* [ Proxy-Info ]
* [ AVP ]

```

Figure 10

4. Diameter NAS Application AVPs

The following sections define a new derived AVP data format, a set of application-specific AVPs and describe the use of AVPs defined in other documents by the Diameter NAS Application.

4.1. Derived AVP Data Formats

4.1.1. QoSFilterRule

The QoSFilterRule format is derived from the OctetString AVP Base Format. It uses the ASCII charset. Packets may be marked or metered based on the following information:

- o Direction (in or out)
- o Source and destination IP address (possibly masked)

- o Protocol
- o Source and destination port (lists or ranges)
- o DSCP values (no mask or range)

Rules for the appropriate direction are evaluated in order; the first matched rule terminates the evaluation. Each packet is evaluated once. If no rule matches, the packet is treated as best effort. An access device unable to interpret or apply a QoS rule SHOULD NOT terminate the session.

QoSFilterRule filters MUST follow the following format:

```
action dir proto from src to dst [options]
```

where

action

tag Mark packet with a specific DSCP [RFC2474]

meter Meter traffic

dir The format is as described under IPFilterRule [RFC6733]

proto The format is as described under IPFilterRule [RFC6733]

src and dst The format is as described under IPFilterRule [RFC6733]

The options are described in Section 4.4.9.

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

4.2. NAS Session AVPs

Diameter reserves the AVP Codes 0 - 255 for RADIUS Attributes that are implemented in Diameter.

4.2.1. Call and Session Information

This section describes the AVPs specific to Diameter applications that are needed to identify the call and session context and status information. On a request, this information allows the server to qualify the session.

These AVPs are used in addition to the following AVPs from the base protocol specification [RFC6733]:

Session-Id
Auth-Application-Id
Origin-Host
Origin-Realm
Auth-Request-Type
Termination-Cause

The following table gives the possible flag values for the session level AVPs.

		+-----+ AVP Flag Rules +-----+	
		MUST	MUST NOT
Attribute Name	Section Defined		
NAS-Port	4.2.2	M	V
NAS-Port-Id	4.2.3	M	V
NAS-Port-Type	4.2.4	M	V
Called-Station-Id	4.2.5	M	V
Calling-Station-Id	4.2.6	M	V
Connect-Info	4.2.7	M	V
Originating-Line-Info	4.2.8	M	V
Reply-Message	4.2.9	M	V

4.2.2. NAS-Port AVP

The NAS-Port AVP (AVP Code 5) is of type Unsigned32 and contains the physical or virtual port number of the NAS which is authenticating the user. Note that "port" is meant in its sense as a service connection on the NAS, not as an IP protocol identifier, and hence the format and contents of the string that identifies the port are specific to the NAS implementation.

Either the NAS-Port AVP or the NAS-Port-Id AVP (Section 4.2.3) SHOULD be present in the AA-Request (AAR, Section 3.1) command if the NAS differentiates among its ports.

4.2.3. NAS-Port-Id AVP

The NAS-Port-Id AVP (AVP Code 87) is of type UTF8String and consists of 7-bit ASCII text identifying the port of the NAS authenticating the user. Note that "port" is meant in its sense as a service connection on the NAS, not as an IP protocol identifier.

Either the NAS-Port-Id AVP or the NAS-Port AVP (Section 4.2.2) SHOULD be present in the AA-Request (AAR, Section 3.1) command if the NAS differentiates among its ports. NAS-Port-Id is intended for use by NASes that cannot conveniently number their ports.

4.2.4. NAS-Port-Type AVP

The NAS-Port-Type AVP (AVP Code 61) is of type Enumerated and contains the type of the port on which the NAS is authenticating the user. This AVP SHOULD be present if the NAS uses the same NAS-Port number ranges for different service types concurrently.

The currently supported values of the NAS-Port-Type AVP are listed in [RADIUSAttrVals].

4.2.5. Called-Station-Id AVP

The Called-Station-Id AVP (AVP Code 30) is of type UTF8String contains a 7-bit ASCII string sent by the NAS to describe the Layer 2 address the user contacted in the request. For dialup access, this can be a phone number obtained by using the Dialed Number Identification Service (DNIS) or a similar technology. Note that this may be different from the phone number the call comes in on. For use with IEEE 802 access, the Called-Station-Id MAY contain a MAC address formatted as described in Congdon, et al. [RFC3580].

If the Called-Station-Id AVP is present in an AAR message, Auth-Request-Type AVP is set to AUTHORIZE_ONLY and the User-Name AVP is absent, the Diameter Server MAY perform authorization based on this AVP. This can be used by a NAS to request whether a call should be answered based on the DNIS result.

Further codification of this field's allowed content and usage is outside the scope of this specification.

4.2.6. Calling-Station-Id AVP

The Calling-Station-Id AVP (AVP Code 31) is of type UTF8String and contains a 7-bit ASCII string sent by the NAS to describe the Layer 2 address from which the user connected in the request. For dialup access, this is the phone number the call came from, using Automatic Number Identification (ANI) or a similar technology. For use with IEEE 802 access, the Calling-Station-Id AVP MAY contain a MAC address, formatted as described in RFC 3580.

If the Calling-Station-Id AVP is present in an AAR message, the Auth-Request-Type AVP is set to AUTHORIZE_ONLY and the User-Name AVP is absent, the Diameter Server MAY perform authorization based on the value of this AVP. This can be used by a NAS to request whether a call should be answered based on the Layer 2 address (ANI, MAC Address, etc.)

Further codification of this field's allowed content and usage is outside the scope of this specification.

4.2.7. Connect-Info AVP

The Connect-Info AVP (AVP Code 77) is of type UTF8String and is sent in the AA-Request message or an ACR message with the value of the Accounting-Record-Type AVP set to STOP. When sent in the AA-Request, it indicates the nature of the user's connection. The connection speed SHOULD be included at the beginning of the first Connect-Info AVP in the message. If the transmit and receive connection speeds differ, both may be included in the first AVP with the transmit speed listed first (the speed at which the NAS modem transmits), then a slash (/), then the receive speed, and then other optional information.

For example: "28800 V42BIS/LAPM" or "52000/31200 V90"

If sent in an ACR message with the value of the Accounting-Record-Type AVP set to STOP, this attribute may summarize statistics relating to session quality. For example, in IEEE 802.11, the Connect-Info AVP may contain information on the number of link layer retransmissions. The exact format of this attribute is implementation specific.

4.2.8. Originating-Line-Info AVP

The Originating-Line-Info AVP (AVP Code 94) is of type OctetString and is sent by the NAS system to convey information about the origin of the call from an SS7 system.

The Originating Line Information (OLI) element indicates the nature and/or characteristics of the line from which a call originated

(e.g., pay phone, hotel, cellular). Telephone companies are starting to offer OLI to their customers as an option over Primary Rate Interface (PRI). Internet Service Providers (ISPs) can use OLI in addition to Called-Station-Id and Calling-Station-Id attributes to differentiate customer calls and to define different services.

The Value field contains two octets (00 - 99). ANSI T1.113 and BELLCORE 394 can be used for additional information about these values and their use. For information on the currently assigned values, see [ANITypes].

4.2.9. Reply-Message AVP

The Reply-Message AVP (AVP Code 18) is of type UTF8String and contains text that MAY be displayed to the user. When used in an AA-Answer message with a successful Result-Code AVP, it indicates success. When found in an AAA message with a Result-Code other than DIAMETER_SUCCESS, the AVP contains a failure message.

The Reply-Message AVP MAY contain text to prompt the user before another AA-Request attempt. When used in an AA-Answer message containing a Result-Code AVP with the value DIAMETER_MULTI_ROUND_AUTH or in an Re-Auth-Request message, it MAY contain text to prompt the user for a response.

4.3. NAS Authentication AVPs

This section defines the AVPs necessary to carry the authentication information in the Diameter protocol. The functionality defined here provides a RADIUS-like AAA service [RFC2865] over a more reliable and secure transport, as defined in the base protocol [RFC6733].

The following table gives the possible flag values for the session level AVPs.

Attribute Name	Section Defined	AVP Flag rules	
		MUST	MUST NOT
User-Password	4.3.1	M	V
Password-Retry	4.3.2	M	V
Prompt	4.3.3	M	V
CHAP-Auth	4.3.4	M	V
CHAP-Algorithm	4.3.5	M	V
CHAP-Ident	4.3.6	M	V

CHAP-Response	4.3.7	M	V
CHAP-Challenge	4.3.8	M	V
ARAP-Password	4.3.9	M	V
ARAP-Challenge-Response	4.3.10	M	V
ARAP-Security	4.3.11	M	V
ARAP-Security-Data	4.3.12	M	V
-----		-----+-----	

4.3.1. User-Password AVP

The User-Password AVP (AVP Code 2) is of type OctetString and contains the password of the user to be authenticated, or the user's input in a multi-round authentication exchange.

The User-Password AVP contains a user password or one-time password and therefore represents sensitive information. As required by Fajardo, et al. [RFC6733], Diameter messages are encrypted by using IPsec [RFC4301] or TLS [RFC5246]. Unless this AVP is used for one-time passwords, the User-Password AVP SHOULD NOT be used in untrusted proxy environments without encrypting it by using end-to-end security techniques.

The clear-text password (prior to encryption) MUST NOT be longer than 128 bytes in length.

4.3.2. Password-Retry AVP

The Password-Retry AVP (AVP Code 75) is of type Unsigned32 and MAY be included in the AA-Answer if the Result-Code indicates an authentication failure. The value of this AVP indicates how many authentication attempts a user is permitted before being disconnected. This AVP is primarily intended for use when the Framed-Protocol AVP (Section 4.4.10.1) is set to ARAP.

4.3.3. Prompt AVP

The Prompt AVP (AVP Code 76) is of type Enumerated and MAY be present in the AA-Answer message. When present, it is used by the NAS to determine whether the user's response, when entered, should be echoed.

The supported values are listed in [RADIUSAttrVals]

4.3.4. CHAP-Auth AVP

The CHAP-Auth AVP (AVP Code 402) is of type Grouped and contains the information necessary to authenticate a user using the PPP Challenge-Handshake Authentication Protocol (CHAP) [RFC1994]. If the CHAP-Auth AVP is found in a message, the CHAP-Challenge AVP (Section 4.3.8) MUST be present as well. The optional AVPs containing the CHAP response depend upon the value of the CHAP-Algorithm AVP (Section 4.3.8). The grouped AVP has the following ABNF grammar:

```
CHAP-Auth ::= < AVP Header: 402 >
              { CHAP-Algorithm }
              { CHAP-Ident }
              [ CHAP-Response ]
              * [ AVP ]
```

4.3.5. CHAP-Algorithm AVP

The CHAP-Algorithm AVP (AVP Code 403) is of type Enumerated and contains the algorithm identifier used in the computation of the CHAP response [RFC1994]. The following values are currently supported:

CHAP with MD5 5

The CHAP response is computed by using the procedure described in [RFC1994] This algorithm requires that the CHAP-Response AVP (Section 4.3.7) MUST be present in the CHAP-Auth AVP (Section 4.3.4).

4.3.6. CHAP-Ident AVP

The CHAP-Ident AVP (AVP Code 404) is of type OctetString and contains the 1 octet CHAP Identifier used in the computation of the CHAP response [RFC1994]

4.3.7. CHAP-Response AVP

The CHAP-Response AVP (AVP Code 405) is of type OctetString and contains the 16 octet authentication data provided by the user in response to the CHAP challenge [RFC1994].

4.3.8. CHAP-Challenge AVP

The CHAP-Challenge AVP (AVP Code 60) is of type OctetString and contains the CHAP Challenge sent by the NAS to the CHAP peer [RFC1994].

4.3.9. ARAP-Password AVP

The ARAP-Password AVP (AVP Code 70) is of type OctetString and is only present when the Framed-Protocol AVP (Section 4.4.10.1) is included in the message and is set to ARAP. This AVP MUST NOT be present if either the User-Password or the CHAP-Auth AVP is present. See Rigney, et al. [RFC2869] for more information on the contents of this AVP.

4.3.10. ARAP-Challenge-Response AVP

The ARAP-Challenge-Response AVP (AVP Code 84) is of type OctetString and is only present when the Framed-Protocol AVP (Section 4.4.10.1) is included in the message and is set to ARAP. This AVP contains an 8 octet response to the dial-in client's challenge. The Diameter server calculates this value by taking the dial-in client's challenge from the high-order 8 octets of the ARAP-Password AVP and performing DES encryption on this value with the authenticating user's password as the key. If the user's password is fewer than 8 octets in length, the password is padded at the end with NULL octets to a length of 8 before it is used as a key.

4.3.11. ARAP-Security AVP

The ARAP-Security AVP (AVP Code 73) is of type Unsigned32 and MAY be present in the AA-Answer message if the Framed-Protocol AVP (Section 4.4.10.1) is set to the value of ARAP, and the Result-Code AVP ([RFC6733], Section 7.1) is set to DIAMETER_MULTI_ROUND_AUTH. See RFC 2869 for more information on the contents of this AVP.

4.3.12. ARAP-Security-Data AVP

The ARAP-Security-Data AVP (AVP Code 74) is of type OctetString and MAY be present in the AA-Request or AA-Answer message if the Framed-Protocol AVP (Section 4.4.10.1) is set to the value of ARAP and the Result-Code AVP ([RFC6733], Section 7.1) is set to DIAMETER_MULTI_ROUND_AUTH. This AVP contains the security module challenge or response associated with the ARAP Security Module specified in the ARAP-Security AVP (Section 4.3.11).

4.4. NAS Authorization AVPs

This section contains the authorization AVPs supported in the NAS Application. The Service-Type AVP SHOULD be present in all messages and, based on its value, additional AVPs defined in this section and Section 4.5 MAY be present.

The following table gives the possible flag values for the session-level AVPs.

Attribute Name	Section Defined	AVP Flag rules	
		MUST	MUST NOT
Service-Type	4.4.1	M	V
Callback-Number	4.4.2	M	V
Callback-Id	4.4.3	M	V
Idle-Timeout	4.4.4	M	V
Port-Limit	4.4.5	M	V
NAS-Filter-Rule	4.4.6	M	V
Filter-Id	4.4.7	M	V
Configuration-Token	4.4.8	M	V
QoS-Filter-Rule	4.4.9		
Framed-Protocol	4.4.10.1	M	V
Framed-Routing	4.4.10.2	M	V
Framed-MTU	4.4.10.3	M	V
Framed-Compression	4.4.10.4	M	V
Framed-IP-Address	4.4.10.5.1	M	V
Framed-IP-Netmask	4.4.10.5.2	M	V
Framed-Route	4.4.10.5.3	M	V
Framed-Pool	4.4.10.5.4	M	V
Framed-Interface-Id	4.4.10.5.5	M	V
Framed-IPv6-Prefix	4.4.10.5.6	M	V
Framed-IPv6-Route	4.4.10.5.7	M	V
Framed-IPv6-Pool	4.4.10.5.8	M	V
Framed-IPX-Network	4.4.10.6.1	M	V
Framed-Appletalk-Link	4.4.10.7.1	M	V
Framed-Appletalk-Network	4.4.10.7.2	M	V
Framed-Appletalk-Zone	4.4.10.7.3	M	V
ARAP-Features	4.4.10.8.1	M	V
ARAP-Zone-Access	4.4.10.8.2	M	V
Login-IP-Host	4.4.11.1	M	V
Login-IPv6-Host	4.4.11.2	M	V
Login-Service	4.4.11.3	M	V
Login-TCP-Port	4.4.11.4.1	M	V
Login-LAT-Service	4.4.11.5.1	M	V
Login-LAT-Node	4.4.11.5.2	M	V
Login-LAT-Group	4.4.11.5.3	M	V
Login-LAT-Port	4.4.11.5.4	M	V

4.4.1.1. Service-Type AVP

The Service-Type AVP (AVP Code 6) is of type Enumerated and contains the type of service the user has requested or the type of service to be provided. One such AVP MAY be present in an authentication and/or authorization request or response. A NAS is not required to implement all of these service types. It MUST treat unknown or unsupported Service-Types received in a response as a failure and end the session with a DIAMETER_INVALID_AVP_VALUE Result-Code.

When used in a request, the Service-Type AVP SHOULD be considered a hint to the server that the NAS believes the user would prefer the kind of service indicated. The server is not required to honor the hint. Furthermore, if the service specified by the server is supported, but not compatible with the current mode of access, the NAS MUST fail to start the session. The NAS MUST also generate the appropriate error message(s).

The complete list of defined values that the Service-Type AVP can take can be found in Rigney, et al. [RFC2865] and the relevant IANA registry [RADIUSAttrVals], but the following values require further qualification here:

Login (1)

The user should be connected to a host. The message MAY include additional AVPs as defined in Section 4.4.11.4 or Section 4.4.11.5.

Framed (2)

A Framed Protocol, such as PPP or SLIP, should be started for the User. The message MAY include additional AVPs defined in Section 4.4.10, or Section 4.5 for tunneling services.

Callback Login (3)

The user should be disconnected and called back, then connected to a host. The message MAY include additional AVPs defined in this Section.

Callback Framed (4)

The user should be disconnected and called back, and then a Framed Protocol, such as PPP or SLIP, should be started for the user. The message MAY include additional AVPs defined in Section 4.4.10, or Section 4.5 for tunneling services.

4.4.2. Callback-Number AVP

The Callback-Number AVP (AVP Code 19) is of type UTF8String and contains a dialing string to be used for callback, the format of which is deployment-specific. The Callback-Number AVP MAY be used in an authentication and/or authorization request as a hint to the server that a callback service is desired, but the server is not required to honor the hint in the corresponding response.

Any further codification of this field's allowed usage range is outside the scope of this specification.

4.4.3. Callback-Id AVP

The Callback-Id AVP (AVP Code 20) is of type UTF8String and contains the name of a place to be called, to be interpreted by the NAS. This AVP MAY be present in an authentication and/or authorization response.

This AVP is not roaming-friendly as it assumes that the Callback-Id is configured on the NAS. Using the Callback-Number AVP (Section 4.4.2) is therefore RECOMMENDED.

4.4.4. Idle-Timeout AVP

The Idle-Timeout AVP (AVP Code 28) is of type Unsigned32 and sets the maximum number of consecutive seconds of idle connection allowable to the user before termination of the session or before a prompt is issued. The default is none, or system specific.

4.4.5. Port-Limit AVP

The Port-Limit AVP (AVP Code 62) is of type Unsigned32 and sets the maximum number of ports the NAS provides to the user. It MAY be used in an authentication and/or authorization request as a hint to the server that multilink PPP [RFC1990] service is desired, but the server is not required to honor the hint in the corresponding response.

4.4.6. NAS-Filter-Rule AVP

The NAS-Filter-Rule AVP (AVP Code 400) is of type IPFilterRule and provides filter rules that need to be configured on the NAS for the user. One or more of these AVPs MAY be present in an authorization response.

4.4.7. Filter-Id AVP

The Filter-Id AVP (AVP Code 11) is of type UTF8String and contains the name of the filter list for this user. It is intended to be human-readable. Zero or more Filter-Id AVPs MAY be sent in an authorization answer message.

Identifying a filter list by name allows the filter to be used on different NASes without regard to filter-list implementation details. However, this AVP is not roaming-friendly, as filter naming differs from one service provider to another.

In environments where backward compatibility with RADIUS is not required, it is RECOMMENDED that the NAS-Filter-Rule AVP (Section 4.4.6) be used instead.

4.4.8. Configuration-Token AVP

The Configuration-Token AVP (AVP Code 78) is of type OctetString and is sent by a Diameter Server to a Diameter Proxy Agent in an AA-Answer command to indicate a type of user profile to be used. It should not be sent to a Diameter Client (NAS).

The format of the Data field of this AVP is site specific.

4.4.9. QoS-Filter-Rule AVP

The QoS-Filter-Rule AVP (AVP Code 407) is of type QoSFilterRule (Section 4.1.1) and provides QoS filter rules that need to be configured on the NAS for the user. One or more such AVPs MAY be present in an authorization response.

The use of this AVP is NOT RECOMMENDED; the AVPs defined by Korhonen, et al. [RFC5777] SHOULD be used instead.

The following options are defined for the QoSFilterRule filters:

DSCP <color>

If action is set to tag (Section 4.1.1) this option MUST be included in the rule.

Color values are defined in Nichols, et al. [RFC2474]. Exact matching of DSCP values is required (no masks or ranges).

metering <rate> <color_under> <color_over>

The metering option provides Assured Forwarding, as defined in Heinanen, et al. [RFC2597]. and MUST be present if the action is set to meter (Section 4.1.1) The rate option is the throughput, in bits per second, used by the access device to mark packets. Traffic over the rate is marked with the color_over codepoint, and traffic under the rate is marked with the color_under codepoint. The color_under and color_over options contain the drop preferences and MUST conform to the recommended codepoint keywords described in RFC 2597 (e.g., AF13).

The metering option also supports the strict limit on traffic required by Expedited Forwarding, as defined in Davie, et al. [RFC3246]. The color_over option may contain the keyword "drop" to prevent forwarding of traffic that exceeds the rate parameter.

4.4.10. Framed Access Authorization AVPs

This section lists the authorization AVPs necessary to support framed access, such as PPP and SLIP. AVPs defined in this section MAY be present in a message if the Service-Type AVP was set to "Framed" or "Callback Framed".

4.4.10.1. Framed-Protocol AVP

The Framed-Protocol AVP (AVP Code 7) is of type Enumerated and contains the framing to be used for framed access. This AVP MAY be present in both requests and responses. The supported values are listed in [RADIUSAttrVals].

4.4.10.2. Framed-Routing AVP

The Framed-Routing AVP (AVP Code 10) is of type Enumerated and contains the routing method for the user when the user is a router to a network. This AVP SHOULD only be present in authorization responses. The supported values are listed in [RADIUSAttrVals].

4.4.10.3. Framed-MTU AVP

The Framed-MTU AVP (AVP Code 12) is of type Unsigned32 and contains the Maximum Transmission Unit (MTU) to be configured for the user, when it is not negotiated by some other means (such as PPP). This AVP SHOULD only be present in authorization responses. The MTU value MUST be in the range from 64 to 65535.

4.4.10.4. Framed-Compression AVP

The Framed-Compression AVP (AVP Code 13) is of type Enumerated and contains the compression protocol to be used for the link. It MAY be used in an authorization request as a hint to the server that a specific compression type is desired, but the server is not required to honor the hint in the corresponding response.

More than one compression protocol AVP MAY be sent. The NAS is responsible for applying the proper compression protocol to the appropriate link traffic.

The supported values are listed in [RADIUSAttrVals].

4.4.10.5. IP Access Authorization AVPs

The AVPs defined in this section are used when the user requests, or is being granted, access service to IP.

4.4.10.5.1. Framed-IP-Address AVP

The Framed-IP-Address AVP (AVP Code 8) [RFC2865] is of type OctetString and contains an IPv4 address of the type specified in the attribute value to be configured for the user. It MAY be used in an authorization request as a hint to the server that a specific address is desired, but the server is not required to honor the hint in the corresponding response.

Two values have special significance: 0xFFFFFFFF and 0xFFFFFFFFE. The value 0xFFFFFFFF indicates that the NAS should allow the user to select an address (i.e., negotiated). The value 0xFFFFFFFFE indicates that the NAS should select an address for the user (e.g., assigned from a pool of addresses kept by the NAS).

4.4.10.5.2. Framed-IP-Netmask AVP

The Framed-IP-Netmask AVP (AVP Code 9) is of type OctetString and contains the four octets of the IPv4 netmask to be configured for the user when the user is a router to a network. It MAY be used in an authorization request as a hint to the server that a specific netmask

is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST be present in a response if the request included this AVP with a value of 0xFFFFFFFF.

4.4.10.5.3. Framed-Route AVP

The Framed-Route AVP (AVP Code 22) is of type UTF8String and contains the 7-bit ASCII routing information to be configured for the user on the NAS. Zero or more of these AVPs MAY be present in an authorization response.

The string MUST contain a destination prefix in dotted quad form optionally followed by a slash and a decimal length specifier stating how many high-order bits of the prefix should be used. This is followed by a space, a gateway address in dotted quad form, a space, and one or more metrics separated by spaces; for example,

```
"192.0.2.0/24 192.0.2.1 1"
```

The length specifier may be omitted, in which case it should default to 8 bits for class A prefixes, to 16 bits for class B prefixes, and to 24 bits for class C prefixes; for example,

```
"192.0.2.0 192.0.2.1 1"
```

Whenever the gateway address is specified as "0.0.0.0" the IP address of the user SHOULD be used as the gateway address.

4.4.10.5.4. Framed-Pool AVP

The Framed-Pool AVP (AVP Code 88) is of type OctetString and contains the name of an assigned address pool that SHOULD be used to assign an address for the user. If a NAS does not support multiple address pools, the NAS SHOULD ignore this AVP. Address pools are usually used for IP addresses but can be used for other protocols if the NAS supports pools for those protocols.

Although specified as type OctetString for compatibility with RADIUS [RFC2869], the encoding of the Data field SHOULD also conform to the rules for the UTF8String Data Format.

4.4.10.5.5. Framed-Interface-Id AVP

The Framed-Interface-Id AVP (AVP Code 96) is of type Unsigned64 and contains the IPv6 interface identifier to be configured for the user. It MAY be used in authorization requests as a hint to the server that a specific interface id is desired, but the server is not required to honor the hint in the corresponding response.

4.4.10.5.6. Framed-IPv6-Prefix AVP

The Framed-IPv6-Prefix AVP (AVP Code 97) is of type OctetString and contains the IPv6 prefix to be configured for the user. One or more AVPs MAY be used in authorization requests as a hint to the server that specific IPv6 prefixes are desired, but the server is not required to honor the hint in the corresponding response.

4.4.10.5.7. Framed-IPv6-Route AVP

The Framed-IPv6-Route AVP (AVP Code 99) is of type UTF8String and contains the ASCII routing information to be configured for the user on the NAS. Zero or more of these AVPs MAY be present in an authorization response.

The string MUST contain an IPv6 address prefix followed by a slash and a decimal length specifier stating how many high order bits of the prefix should be used. This is followed by a space, a gateway address in hexadecimal notation, a space, and one or more metrics separated by spaces; for example,

```
"2001:db8::/32 2001:db8:106:a00:20ff:fe99:a998 1"
```

Whenever the gateway address is the IPv6 unspecified address, the IP address of the user SHOULD be used as the gateway address, such as in:

```
"2001:db8::/32 :: 1"
```

4.4.10.5.8. Framed-IPv6-Pool AVP

The Framed-IPv6-Pool AVP (AVP Code 100) is of type OctetString and contains the name of an assigned pool that SHOULD be used to assign an IPv6 prefix for the user. If the access device does not support multiple prefix pools, it MUST ignore this AVP.

Although specified as type OctetString for compatibility with RADIUS [RFC3162], the encoding of the Data field SHOULD also conform to the rules for the UTF8String Data Format.

4.4.10.6. IPX Access AVPs

The AVPs defined in this section are used when the user requests, or is being granted, access to an IPX network service [IPX].

4.4.10.6.1. Framed-IPX-Network AVP

The Framed-IPX-Network AVP (AVP Code 23) is of type Unsigned32 and contains the IPX Network number to be configured for the user. It MAY be used in an authorization request as a hint to the server that a specific address is desired, but the server is not required to honor the hint in the corresponding response.

Two addresses have special significance: 0xFFFFFFFF and 0xFFFFFFFFE. The value 0xFFFFFFFF indicates that the NAS should allow the user to select an address (i.e., Negotiated). The value 0xFFFFFFFFE indicates that the NAS should select an address for the user (e.g., assign it from a pool of one or more IPX networks kept by the NAS).

4.4.10.7. AppleTalk Network Access AVPs

The AVPs defined in this section are used when the user requests, or is being granted, access to an AppleTalk network [AppleTalk].

4.4.10.7.1. Framed-AppleTalk-Link AVP

The Framed-AppleTalk-Link AVP (AVP Code 37) is of type Unsigned32 and contains the AppleTalk network number that should be used for the serial link to the user, which is another AppleTalk router. This AVP MUST only be present in an authorization response and is never used when the user is not another router.

Despite the size of the field, values range from 0 to 65,535. The special value of 0 indicates an unnumbered serial link. A value of 1 to 65,535 means that the serial line between the NAS and the user should be assigned that value as an AppleTalk network number.

4.4.10.7.2. Framed-AppleTalk-Network AVP

The Framed-AppleTalk-Network AVP (AVP Code 38) is of type Unsigned32 and contains the AppleTalk Network number that the NAS should probe to allocate an AppleTalk node for the user. This AVP MUST only be present in an authorization response and is never used when the user is not another router. Multiple instances of this AVP indicate that the NAS may probe, using any of the network numbers specified.

Despite the size of the field, values range from 0 to 65,535. The special value 0 indicates that the NAS should assign a network for the user, using its default cable range. A value between 1 and 65,535 (inclusive) indicates to the AppleTalk Network that the NAS should probe to find an address for the user.

4.4.10.7.3. Framed-AppleTalk-Zone AVP

The Framed-AppleTalk-Zone AVP (AVP Code 39) is of type OctetString and contains the AppleTalk Default Zone to be used for this user. This AVP MUST only be present in an authorization response. Multiple instances of this AVP in the same message are not allowed.

The codification of this field's allowed range is outside the scope of this specification.

4.4.10.8. AppleTalk Remote Access AVPs

The AVPs defined in this section are used when the user requests, or is being granted, access to the AppleTalk network via the AppleTalk Remote Access Protocol [ARAP]. They are only present if the Framed-Protocol AVP (Section 4.4.10.1) is set to ARAP. Section 2.2 of RFC 2869 describes the operational use of these attributes.

4.4.10.8.1. ARAP-Features AVP

The ARAP-Features AVP (AVP Code 71) is of type OctetString and MAY be present in the AA-Accept message if the Framed-Protocol AVP is set to the value of ARAP. See RFC 2869 for more information about the format of this AVP.

4.4.10.8.2. ARAP-Zone-Access AVP

The ARAP-Zone-Access AVP (AVP Code 72) is of type Enumerated and MAY be present in the AA-Accept message if the Framed-Protocol AVP is set to the value of ARAP.

The supported values are listed in [RADIUSAttrVals] and defined in RFC 2869.

4.4.11. Non-Framed Access Authorization AVPs

This section contains the authorization AVPs that are needed to support terminal server functionality. AVPs defined in this section MAY be present in a message if the Service-Type AVP was set to "Login" or "Callback Login".

4.4.11.1. Login-IP-Host AVP

The Login-IP-Host AVP (AVP Code 14) [RFC2865] is of type OctetString and contains the IPv4 address of a host with which to connect the user when the Login-Service AVP is included. It MAY be used in an AA-Request command as a hint to the Diameter Server that a specific host is desired, but the Diameter Server is not required to honor the hint in the AA-Answer.

Two addresses have special significance: all ones and 0. The value of all ones indicates that the NAS SHOULD allow the user to select an address. The value 0 indicates that the NAS SHOULD select a host to connect the user to.

4.4.11.2. Login-IPv6-Host AVP

The Login-IPv6-Host AVP (AVP Code 98) [RFC3162] is of type OctetString and contains the IPv6 address of a host with which to connect the user when the Login-Service AVP is included. It MAY be used in an AA-Request command as a hint to the Diameter Server that a specific host is desired, but the Diameter Server is not required to honor the hint in the AA-Answer.

Two addresses have special significance, 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF and 0. The value 0xFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFFF indicates that the NAS SHOULD allow the user to select an address. The value 0 indicates that the NAS SHOULD select a host to connect the user to.

4.4.11.3. Login-Service AVP

The Login-Service AVP (AVP Code 15) is of type Enumerated and contains the service that should be used to connect the user to the login host. This AVP SHOULD only be present in authorization responses. The supported values are listed in RFC 2869.

4.4.11.4. TCP Services

The AVP described in the following section MAY be present if the Login-Service AVP is set to Telnet, Rlogin, TCP Clear, or TCP Clear Quiet.

4.4.11.4.1. Login-TCP-Port AVP

The Login-TCP-Port AVP (AVP Code 16) is of type Unsigned32 and contains the TCP port with which the user is to be connected when the Login-Service AVP is also present. This AVP SHOULD only be present in authorization responses. The value MUST NOT be greater than 65,535.

4.4.11.5. LAT Services

The AVPs described in this section MAY be present if the Login-Service AVP is set to LAT [LAT].

4.4.11.5.1. Login-LAT-Service AVP

The Login-LAT-Service AVP (AVP Code 34) is of type OctetString and contains the system with which the user is to be connected by LAT. It MAY be used in an authorization request as a hint to the server that a specific service is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST only be present in the response if the Login-Service AVP states that LAT is desired.

Administrators use this service attribute when dealing with clustered systems. In these environments, several different time-sharing hosts share the same resources (disks, printers, etc.), and administrators often configure each host to offer access (service) to each of the shared resources. In this case, each host in the cluster advertises its services through LAT broadcasts.

Sophisticated users often know which service providers (machines) are faster and tend to use a node name when initiating a LAT connection. Some administrators want particular users to use certain machines as a primitive form of load balancing (although LAT knows how to do load balancing itself).

The String field contains the identity of the LAT service to use. The LAT Architecture allows this string to contain \$ (dollar), - (hyphen), . (period), _ (underscore), numerics, upper- and lowercase alphabets, and the ISO Latin-1 character set extension [ISO.8859-1.1987]. All LAT string comparisons are case insensitive.

4.4.11.5.2. Login-LAT-Node AVP

The Login-LAT-Node AVP (AVP Code 35) is of type OctetString and contains the Node with which the user is to be automatically connected by LAT. It MAY be used in an authorization request as a hint to the server that a specific LAT node is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST only be present in a response if the Login-Service-Type AVP is set to LAT.

The String field contains the identity of the LAT service to use. The LAT Architecture allows this string to contain \$ (dollar), - (hyphen), . (period), _ (underscore), numerics, upper- and lowercase alphabets, and the ISO Latin-1 character set extension [ISO.8859-1.1987]. All LAT string comparisons are case insensitive.

4.4.11.5.3. Login-LAT-Group AVP

The Login-LAT-Group AVP (AVP Code 36) is of type OctetString and contains a string identifying the LAT group codes this user is authorized to use. It MAY be used in an authorization request as a hint to the server that a specific group is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST only be present in a response if the Login-Service-Type AVP is set to LAT.

LAT supports 256 different group codes, which LAT uses as a form of access rights. LAT encodes the group codes as a 256-bit bitmap.

Administrators can assign one or more of the group code bits at the LAT service provider; it will only accept LAT connections that have these group codes set in the bitmap. The administrators assign a bitmap of authorized group codes to each user. LAT gets these from the operating system and uses them in its requests to the service providers.

The codification of the range of allowed usage of this field is outside the scope of this specification.

4.4.11.5.4. Login-LAT-Port AVP

The Login-LAT-Port AVP (AVP Code 63) is of type OctetString and contains the Port with which the user is to be connected by LAT. It MAY be used in an authorization request as a hint to the server that a specific port is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST only be present in a response if the Login-Service-Type AVP is set to LAT.

The String field contains the identity of the LAT service to use. The LAT Architecture allows this string to contain \$ (dollar), - (hyphen), . (period), _ (underscore), numerics, upper- and lower-case alphabets, and the ISO Latin-1 character set extension [ISO.8859-1.1987].

All LAT string comparisons are case insensitive.

4.5. NAS Tunneling AVPs

Some NASes support compulsory tunnel services in which the incoming connection data is conveyed by an encapsulation method to a gateway elsewhere in the network. This is typically transparent to the service user, and the tunnel characteristics may be described by the remote AAA server, based on the user's authorization information. Several tunnel characteristics may be returned, and the NAS

implementation may choose one. See Zorn, et al. [RFC2868] and Zorn, Aboba & Mitton [RFC2867] for further information.

The following table gives the possible flag values for the session level AVPs and specifies whether the AVP MAY be encrypted.

		+-----+	
		AVP Flag rules	
		-----+	
Attribute Name	Section Defined	MUST	MUST NOT
		-----	-----
Tunneling	4.5.1	M	V
Tunnel-Type	4.5.2	M	V
Tunnel-Medium-Type	4.5.3	M	V
Tunnel-Client-Endpoint	4.5.4	M	V
Tunnel-Server-Endpoint	4.5.5	M	V
Tunnel-Password	4.5.6	M	V
Tunnel-Private-Group-Id	4.5.7	M	V
Tunnel-Assignment-Id	4.5.8	M	V
Tunnel-Preference	4.5.9	M	V
Tunnel-Client-Auth-Id	4.5.10	M	V
Tunnel-Server-Auth-Id	4.5.11	M	V
-----		-----	-----

4.5.1. Tunneling AVP

The Tunneling AVP (AVP Code 401) is of type Grouped and contains the following AVPs, used to describe a compulsory tunnel service ([RFC2868], [RFC2867]). Its data field has the following ABNF grammar:

```
Tunneling ::= < AVP Header: 401 >
              { Tunnel-Type }
              { Tunnel-Medium-Type }
              { Tunnel-Client-Endpoint }
              { Tunnel-Server-Endpoint }
              [ Tunnel-Preference ]
              [ Tunnel-Client-Auth-Id ]
              [ Tunnel-Server-Auth-Id ]
              [ Tunnel-Assignment-Id ]
              [ Tunnel-Password ]
              [ Tunnel-Private-Group-Id ]
```

4.5.2. Tunnel-Type AVP

The Tunnel-Type AVP (AVP Code 64) is of type Enumerated and contains the tunneling protocol(s) to be used (in the case of a tunnel initiator) or in use (in the case of a tunnel terminator). It MAY be used in an authorization request as a hint to the server that a specific tunnel type is desired, but the server is not required to honor the hint in the corresponding response.

The Tunnel-Type AVP SHOULD also be included in ACR messages.

A tunnel initiator is not required to implement any of these tunnel types. If a tunnel initiator receives a response that contains only unknown or unsupported Tunnel-Types, the tunnel initiator MUST behave as though a response were received with the Result-Code indicating a failure.

The supported values are listed in [RADIUSAttrVals].

4.5.3. Tunnel-Medium-Type AVP

The Tunnel-Medium-Type AVP (AVP Code 65) is of type Enumerated and contains the transport medium to use when creating a tunnel for protocols (such as L2TP [RFC3931]) that can operate over multiple transports. It MAY be used in an authorization request as a hint to the server that a specific medium is desired, but the server is not required to honor the hint in the corresponding response.

The supported values are listed in [RADIUSAttrVals].

4.5.4. Tunnel-Client-Endpoint AVP

The Tunnel-Client-Endpoint AVP (AVP Code 66) is of type UTF8String and contains the address of the initiator end of the tunnel. It MAY be used in an authorization request as a hint to the server that a specific endpoint is desired, but the server is not required to honor the hint in the corresponding response. This AVP SHOULD be included in the corresponding ACR messages, in which case it indicates the address from which the tunnel was initiated. This AVP, along with the Tunnel-Server-Endpoint (Section 4.5.5) and Session-Id AVPs ([RFC6733], Section 8.8), can be used to provide a globally unique means to identify a tunnel for accounting and auditing purposes.

If the value of the Tunnel-Medium-Type AVP (Section 4.5.3) is IPv4 (1), then this string is either the fully qualified domain name (FQDN) of the tunnel client machine, or a "dotted-decimal" IP address. Implementations MUST support the dotted-decimal format and SHOULD support the FQDN format for IP addresses.

If Tunnel-Medium-Type is IPv6 (2), then this string is either the FQDN of the tunnel client machine, or a text representation of the address in either the preferred or alternate form [RFC3516]. Conforming implementations MUST support the preferred form and SHOULD support both the alternate text form and the FQDN format for IPv6 addresses.

If Tunnel-Medium-Type is neither IPv4 nor IPv6, then this string is a tag referring to configuration data local to the Diameter client that describes the interface or medium-specific client address to use.

Note that this application handles internationalized domain names in the same way as the Diameter base protocol (see Appendix D of RFC 6733 for details).

4.5.5. Tunnel-Server-Endpoint AVP

The Tunnel-Server-Endpoint AVP (AVP Code 67) is of type UTF8String and contains the address of the server end of the tunnel. It MAY be used in an authorization request as a hint to the server that a specific endpoint is desired, but the server is not required to honor the hint in the corresponding response.

This AVP SHOULD be included in the corresponding ACR messages, in which case it indicates the address from which the tunnel was initiated. This AVP, along with the Tunnel-Client-Endpoint (Section 4.5.4) and Session-Id AVP ([RFC6733], Section 8.8), can be used to provide a globally unique means to identify a tunnel for accounting and auditing purposes.

If Tunnel-Medium-Type is IPv4 (1), then this string is either the fully qualified domain name (FQDN) of the tunnel server machine, or a "dotted-decimal" IP address. Implementations MUST support the dotted-decimal format and SHOULD support the FQDN format for IP addresses.

If Tunnel-Medium-Type is IPv6 (2), then this string is either the FQDN of the tunnel server machine, or a text representation of the address in either the preferred or alternate form [RFC3516]. Implementations MUST support the preferred form and SHOULD support both the alternate text form and the FQDN format for IPv6 addresses.

If Tunnel-Medium-Type is not IPv4 or IPv6, this string is a tag referring to configuration data local to the Diameter client that describes the interface or medium-specific server address to use.

Note that this application handles internationalized domain names in the same way as the Diameter base protocol (see Appendix D of RFC 6733 for details).

4.5.6. Tunnel-Password AVP

The Tunnel-Password AVP (AVP Code 69) is of type OctetString and may contain a password to be used to authenticate to a remote server.

The Tunnel-Password AVP SHOULD NOT be used in untrusted proxy environments without encrypting it by using end-to-end security techniques.

4.5.7. Tunnel-Private-Group-Id AVP

The Tunnel-Private-Group-Id AVP (AVP Code 81) is of type OctetString and contains the group Id for a particular tunneled session. The Tunnel-Private-Group-Id AVP MAY be included in an authorization request if the tunnel initiator can predetermine the group resulting from a particular connection. It SHOULD be included in the authorization response if this tunnel session is to be treated as belonging to a particular private group. Private groups may be used to associate a tunneled session with a particular group of users. For example, it MAY be used to facilitate routing of unregistered IP addresses through a particular interface. This AVP SHOULD be included in the ACR messages that pertain to the tunneled session.

4.5.8. Tunnel-Assignment-Id AVP

The Tunnel-Assignment-Id AVP (AVP Code 82) is of type OctetString and is used to indicate to the tunnel initiator the particular tunnel to which a session is to be assigned. Some tunneling protocols, such as PPTP [RFC2637] and L2TP [RFC3931], allow for sessions between the same two tunnel endpoints to be multiplexed over the same tunnel and also for a given session to use its own dedicated tunnel. This attribute provides a mechanism for Diameter to inform the tunnel initiator (for example, a LAC) whether to assign the session to a multiplexed tunnel or to a separate tunnel. Furthermore, it allows for sessions sharing multiplexed tunnels to be assigned to different multiplexed tunnels.

A particular tunneling implementation may assign differing characteristics to particular tunnels. For example, different tunnels may be assigned different QoS parameters. Such tunnels may be used to carry either individual or multiple sessions. The Tunnel-Assignment-Id attribute thus allows the Diameter server to indicate that a particular session is to be assigned to a tunnel providing an appropriate level of service. It is expected that any QoS-related

Diameter tunneling attributes defined in the future accompanying this one will be associated by the tunnel initiator with the Id given by this attribute. In the meantime, any semantic given to a particular Id string is a matter left to local configuration in the tunnel initiator.

The Tunnel-Assignment-Id AVP is of significance only to Diameter and the tunnel initiator. The Id it specifies is only intended to be of local use to Diameter and the tunnel initiator. The Id assigned by the tunnel initiator is not conveyed to the tunnel peer.

This attribute MAY be included in authorization responses. The tunnel initiator receiving this attribute MAY choose to ignore it and to assign the session to an arbitrary multiplexed or non-multiplexed tunnel between the desired endpoints. This AVP SHOULD also be included in the Accounting-Request messages pertaining to the tunneled session.

If a tunnel initiator supports the Tunnel-Assignment-Id AVP, then it should assign a session to a tunnel in the following manner:

- o If this AVP is present and a tunnel exists between the specified endpoints with the specified Id, then the session should be assigned to that tunnel.
- o If this AVP is present and no tunnel exists between the specified endpoints with the specified Id, then a new tunnel should be established for the session and the specified Id should be associated with the new tunnel.
- o If this AVP is not present, then the session is assigned to an unnamed tunnel. If an unnamed tunnel does not yet exist between the specified endpoints, then it is established and used for this session and for subsequent ones established without the Tunnel-Assignment-Id attribute. A tunnel initiator MUST NOT assign a session for which a Tunnel-Assignment-Id AVP was not specified to a named tunnel (i.e., one that was initiated by a session specifying this AVP).

Note that the same Id may be used to name different tunnels if these tunnels are between different endpoints.

4.5.9. Tunnel-Preference AVP

The Tunnel-Preference AVP (AVP Code 83) is of type Unsigned32 and is used to identify the relative preference assigned to each tunnel when more than one set of tunneling AVPs is returned within separate Grouped-AVP AVPs. It MAY be used in an authorization request as a

hint to the server that a specific preference is desired, but the server is not required to honor the hint in the corresponding response.

For example, suppose that AVPs describing two tunnels are returned by the server, one with a Tunnel-Type of PPTP and the other with a Tunnel-Type of L2TP. If the tunnel initiator supports only one of the Tunnel-Types returned, it will initiate a tunnel of that type. If, however, it supports both tunnel protocols, it SHOULD use the value of the Tunnel-Preference AVP to decide which tunnel should be started. The tunnel with the lowest numerical value in the Value field of this AVP SHOULD be given the highest preference. The values assigned to two or more instances of the Tunnel-Preference AVP within a given authorization response MAY be identical. In this case, the tunnel initiator SHOULD use locally configured metrics to decide which set of AVPs to use.

4.5.10. Tunnel-Client-Auth-Id AVP

The Tunnel-Client-Auth-Id AVP (AVP Code 90) is of type UTF8String and specifies the 7-bit US-ASCII name used by the tunnel initiator during the authentication phase of tunnel establishment. It MAY be used in an authorization request as a hint to the server that a specific preference is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST be present in the authorization response if an authentication name other than the default is desired. This AVP SHOULD be included in the ACR messages pertaining to the tunneled session.

4.5.11. Tunnel-Server-Auth-Id AVP

The Tunnel-Server-Auth-Id AVP (AVP Code 91) is of type UTF8String and specifies the 7-bit US-ASCII name used by the tunnel terminator during the authentication phase of tunnel establishment. It MAY be used in an authorization request as a hint to the server that a specific preference is desired, but the server is not required to honor the hint in the corresponding response. This AVP MUST be present in the authorization response if an authentication name other than the default is desired. This AVP SHOULD be included in the ACR messages pertaining to the tunneled session.

4.6. NAS Accounting AVPs

Applications implementing this specification use Diameter Accounting (as defined in [RFC6733]) and the AVPs in the following section. Service-specific AVP usage is defined in the tables in Section 5.

If accounting is active, Accounting Request (ACR) messages SHOULD be sent after the completion of any Authentication or Authorization transaction and at the end of a Session. The value of the Accounting-Record-Type AVP [RFC6733] indicates the type of event. All other AVPs identify the session and provide additional information relevant to the event.

The successful completion of the first Authentication or Authorization transaction SHOULD cause a START_RECORD to be sent. If additional Authentications or Authorizations occur in later transactions, the first exchange should generate a START_RECORD, and the later an INTERIM_RECORD. For a given session, there MUST only be one set of matching START and STOP records, with any number of INTERIM_RECORDS in between, or one EVENT_RECORD indicating the reason a session wasn't started.

The following table gives the possible flag values for the session level AVPs and specifies whether the AVP MAY be encrypted.

Attribute Name	Section Defined	AVP Flag rules	
		MUST	MUST NOT
Accounting-Input-Octets	4.6.1	M	V
Accounting-Output-Octets	4.6.2	M	V
Accounting-Input-Packets	4.6.3	M	V
Accounting-Output-Packets	4.6.4	M	V
Acct-Session-Time	4.6.5	M	V
Acct-Authentic	4.6.6	M	V
Accounting-Auth-Method	4.6.7	M	V
Acct-Delay-Time	4.6.8	M	V
Acct-Link-Count	4.6.9	M	V
Acct-Tunnel-Connection	4.6.10	M	V
Acct-Tunnel-Packets-Lost	4.6.11	M	V

4.6.1. Accounting-Input-Octets AVP

The Accounting-Input-Octets AVP (AVP Code 363) is of type Unsigned64 and contains the number of octets received from the user.

For NAS usage, this AVP indicates how many octets have been received from the port in the course of this session. It can only be present in ACR messages with an Accounting-Record-Type [RFC6733] of INTERIM_RECORD or STOP_RECORD.

4.6.2. Accounting-Output-Octets AVP

The Accounting-Output-Octets AVP (AVP Code 364) is of type Unsigned64 and contains the number of octets sent to the user.

For NAS usage, this AVP indicates how many octets have been sent to the port in the course of this session. It can only be present in ACR messages with an Accounting-Record-Type of INTERIM_RECORD or STOP_RECORD.

4.6.3. Accounting-Input-Packets AVP

The Accounting-Input-Packets (AVP Code 365) is of type Unsigned64 and contains the number of packets received from the user.

For NAS usage, this AVP indicates how many packets have been received from the port over the course of a session being provided to a Framed User. It can only be present in ACR messages with an Accounting-Record-Type of INTERIM_RECORD or STOP_RECORD.

4.6.4. Accounting-Output-Packets AVP

The Accounting-Output-Packets (AVP Code 366) is of type Unsigned64 and contains the number of IP packets sent to the user.

For NAS usage, this AVP indicates how many packets have been sent to the port over the course of a session being provided to a Framed User. It can only be present in ACR messages with an Accounting-Record-Type of INTERIM_RECORD or STOP_RECORD.

4.6.5. Acct-Session-Time AVP

The Acct-Session-Time AVP (AVP Code 46) is of type Unsigned32 and indicates the length of the current session in seconds. It can only be present in ACR messages with an Accounting-Record-Type of INTERIM_RECORD or STOP_RECORD.

4.6.6. Acct-Authentic AVP

The Acct-Authentic AVP (AVP Code 45) is of type Enumerated and specifies how the user was authenticated. The supported values are listed in [RADIUSAttrVals].

4.6.7. Accounting-Auth-Method AVP

The Accounting-Auth-Method AVP (AVP Code 406) is of type Enumerated. A NAS MAY include this AVP in an Accounting-Request message to indicate the method used to authenticate the user. (Note that this AVP is semantically equivalent, and the supported values are identical, to the Microsoft MS-Acct-Auth-Type vendor-specific RADIUS attribute [RFC2548]).

4.6.8. Acct-Delay-Time AVP

The Acct-Delay-Time AVP (AVP Code 41) is of type Unsigned32 and indicates the number of seconds the Diameter client has been trying to send the Accounting-Request (ACR). The accounting server may subtract this value from the time when the ACR arrives at the server to calculate the approximate time of the event that caused the ACR to be generated.

This AVP is not used for retransmissions at the transport level (TCP or SCTP). Rather, it may be used when an ACR command cannot be transmitted because there is no appropriate peer to transmit it to or was rejected because it could not be delivered. In these cases, the command MAY be buffered and transmitted later, when an appropriate peer-connection is available or after sufficient time has passed that the destination-host may be reachable and operational. If the ACR is re-sent in this way, the Acct-Delay-Time AVP SHOULD be included. The value of this AVP indicates the number of seconds that elapsed between the time of the first attempt at transmission and the current attempt.

4.6.9. Acct-Link-Count AVP

The Acct-Link-Count AVP (AVP Code 51) is of type Unsigned32 and indicates the total number of links that have been active (current or closed) in a given multilink session at the time the accounting record is generated. This AVP MAY be included in Accounting-Requests for any session that may be part of a multilink service.

The Acct-Link-Count AVP may be used to make it easier for an accounting server to know when it has all the records for a given multilink service. When the number of Accounting-Requests received with Accounting-Record-Type = STOP_RECORD and with the same Acct-Multi-Session-Id and unique Session-Ids equals the largest value of Acct-Link-Count seen in those Accounting-Requests, all STOP_RECORD Accounting-Requests for that multilink service have been received.

The following example, showing eight Accounting-Requests, illustrates how the Acct-Link-Count AVP is used. In the table below, only the

relevant AVPs are shown, although additional AVPs containing accounting information will be present in the Accounting-Requests.

Acct-Multi- Session-Id	Session-Id	Accounting- Record-Type	Acct- Link-Count
"...10"	"...10"	START_RECORD	1
"...10"	"...11"	START_RECORD	2
"...10"	"...11"	STOP_RECORD	2
"...10"	"...12"	START_RECORD	3
"...10"	"...13"	START_RECORD	4
"...10"	"...12"	STOP_RECORD	4
"...10"	"...13"	STOP_RECORD	4
"...10"	"...10"	STOP_RECORD	4

4.6.10. Acct-Tunnel-Connection AVP

The Acct-Tunnel-Connection AVP (AVP Code 68) is of type OctetString and contains the identifier assigned to the tunnel session. This AVP, along with the Tunnel-Client-Endpoint (Section 4.5.4) and Tunnel-Server-Endpoint (Section 4.5.5) AVPs, may be used to provide a means to uniquely identify a tunnel session for auditing purposes.

The format of the identifier in this AVP depends upon the value of the Tunnel-Type AVP (Section 4.5.2). For example, to identify an L2TP tunnel connection fully, the L2TP Tunnel Id and Call Id might be encoded in this field. The exact encoding of this field is implementation dependent.

4.6.11. Acct-Tunnel-Packets-Lost AVP

The Acct-Tunnel-Packets-Lost AVP (AVP Code 86) is of type Unsigned32 and contains the number of packets lost on a given tunnel.

5. AVP Occurrence Tables

The following tables present the AVPs used by NAS applications in NAS messages and specify in which Diameter messages they may or may not be present. Messages and AVPs defined in the base Diameter protocol [RFC6733] are not described in this document. Note that AVPs that can only be present within a Grouped AVP are not represented in this table.

The tables use the following symbols:

- 0 The AVP MUST NOT be present in the message.

- 0+ Zero or more instances of the AVP MAY be present in the message.
- 0-1 Zero or one instance of the AVP MAY be present in the message.
- 1 Exactly one instance of the AVP MUST be present in the message.

5.1. AA-Request/Answer AVP Table

The table in this section is limited to the Command Codes defined in this specification.

AVP Name	Command	
	AAR	AAA
Acct-Interim-Interval	0	0-1
ARAP-Challenge-Response	0	0-1
ARAP-Features	0	0-1
ARAP-Password	0-1	0
ARAP-Security	0-1	0-1
ARAP-Security-Data	0+	0+
ARAP-Zone-Access	0	0-1
Auth-Application-Id	1	1
Auth-Grace-Period	0-1	0-1
Auth-Request-Type	1	1
Auth-Session-State	0-1	0-1
Authorization-Lifetime	0-1	0-1

Attribute Name	Command	
	AAR	AAA
Callback-Id	0	0-1
Callback-Number	0-1	0-1
Called-Station-Id	0-1	0
Calling-Station-Id	0-1	0
CHAP-Auth	0-1	0
CHAP-Challenge	0-1	0
Class	0	0+
Configuration-Token	0	0+
Connect-Info	0+	0
Destination-Host	0-1	0

Destination-Realm	1	0
Error-Message	0	0-1
Error-Reporting-Host	0	0-1
Failed-AVP	0+	0+
Filter-Id	0	0+
Framed-Appletalk-Link	0	0-1
Framed-Appletalk-Network	0	0+
Framed-Appletalk-Zone	0	0-1
Framed-Compression	0+	0+
Framed-Interface-Id	0-1	0-1
Framed-IP-Address	0-1	0-1
Framed-IP-Netmask	0-1	0-1
Framed-IPv6-Prefix	0+	0+
Framed-IPv6-Pool	0	0-1
Framed-IPv6-Route	0	0+
Framed-IPX-Network	0	0-1
Framed-MTU	0-1	0-1
Framed-Pool	0	0-1
Framed-Protocol	0-1	0-1
Framed-Route	0	0+
Framed-Routing	0	0-1
Idle-Timeout	0	0-1
Login-IP-Host	0+	0+
Login-IPv6-Host	0+	0+
Login-LAT-Group	0-1	0-1
Login-LAT-Node	0-1	0-1
Login-LAT-Port	0-1	0-1
Login-LAT-Service	0-1	0-1
Login-Service	0	0-1
Login-TCP-Port	0	0-1
Multi-Round-Time-Out	0	0-1
-----+-----+		

Attribute Name	Command	
	AAR	AAA
-----+-----+		
NAS-Filter-Rule	0	0+
NAS-Identifier	0-1	0
NAS-IP-Address	0-1	0
NAS-IPv6-Address	0-1	0
NAS-Port	0-1	0
NAS-Port-Id	0-1	0
NAS-Port-Type	0-1	0
Origin-AAA-Protocol	0-1	0-1
Origin-Host	1	1
Origin-Realm	1	1

Origin-State-Id	0-1	0-1
Originating-Line-Info	0-1	0
Password-Retry	0	0-1
Port-Limit	0-1	0-1
Prompt	0	0-1
Proxy-Info	0+	0+
QoS-Filter-Rule	0	0+
Re-Auth-Request-Type	0	0-1
Redirect-Host	0	0+
Redirect-Host-Usage	0	0-1
Redirect-Max-Cache-Time	0	0-1
Reply-Message	0	0+
Result-Code	0	1
Route-Record	0+	0
Service-Type	0-1	0-1
Session-Id	1	1
Session-Timeout	0	0-1
State	0-1	0-1
Tunneling	0+	0+
User-Name	0-1	0-1
User-Password	0-1	0
-----	-----	-----

5.2. Accounting AVP Tables

The tables in this section are used to show which AVPs defined in this document are to be present and used in NAS application Accounting messages. These AVPs are defined in this document, as well as in [RFC6733] and [RFC2866].

5.2.1. Framed Access Accounting AVP Table

The table in this section is used when the Service-Type AVP (Section 4.4.1) specifies Framed Access.

Attribute Name	Command	
	ACR	ACA
Accounting-Auth-Method	0-1	0
Accounting-Input-Octets	1	0
Accounting-Input-Packets	1	0
Accounting-Output-Octets	1	0
Accounting-Output-Packets	1	0
Accounting-Record-Number	0-1	0-1
Accounting-Record-Type	1	1
Accounting-Realtime-Required	0-1	0-1

Accounting-Sub-Session-Id	0-1	0-1
Acct-Application-Id	0-1	0-1
Acct-Session-Id	1	0-1
Acct-Multi-Session-Id	0-1	0-1
Acct-Authentic	1	0
Acct-Delay-Time	0-1	0
Acct-Interim-Interval	0-1	0-1
Acct-Link-Count	0-1	0
Acct-Session-Time	1	0
Acct-Tunnel-Connection	0-1	0
Acct-Tunnel-Packets-Lost	0-1	0
Authorization-Lifetime	0-1	0
Callback-Id	0-1	0
Callback-Number	0-1	0
Called-Station-Id	0-1	0
Calling-Station-Id	0-1	0
Class	0+	0+
Connection-Info	0+	0
Destination-Host	0-1	0
Destination-Realm	1	0
Event-Timestamp	0-1	0-1
Error-Message	0	0-1
Error-Reporting-Host	0	0-1
Failed-AVP	0	0+

Attribute Name	Command	
	ACR	ACA
Framed-AppleTalk-Link	0-1	0
Framed-AppleTalk-Network	0-1	0
Framed-AppleTalk-Zone	0-1	0
Framed-Compression	0-1	0
Framed-IP-Address	0-1	0
Framed-IP-Netmask	0-1	0
Framed-IPv6-Prefix	0+	0
Framed-IPv6-Pool	0-1	0
Framed-IPX-Network	0-1	0
Framed-MTU	0-1	0
Framed-Pool	0-1	0
Framed-Protocol	0-1	0
Framed-Route	0-1	0
Framed-Routing	0-1	0
NAS-Filter-Rule	0+	0
NAS-Identifier	0-1	0-1
NAS-IP-Address	0-1	0-1

NAS-IPv6-Address	0-1	0-1
NAS-Port	0-1	0-1
NAS-Port-Id	0-1	0-1
NAS-Port-Type	0-1	0-1
Origin-AAA-Protocol	0-1	0-1
Origin-Host	1	1
Origin-Realm	1	1
Origin-State-Id	0-1	0-1
Originating-Line-Info	0-1	0
Proxy-Info	0+	0+
QoS-Filter-Rule	0+	0
Route-Record	0+	0
Result-Code	0	1
Service-Type	0-1	0-1
Session-Id	1	1
Termination-Cause	0-1	0-1
Tunnel-Assignment-Id	0-1	0
Tunnel-Client-Endpoint	0-1	0
Tunnel-Medium-Type	0-1	0
Tunnel-Private-Group-Id	0-1	0
Tunnel-Server-Endpoint	0-1	0
Tunnel-Type	0-1	0
User-Name	0-1	0-1
-----+-----+		

5.2.2. Non-Framed Access Accounting AVP Table

The table in this section is used when the Service-Type AVP (Section 4.4.1) specifies Non-Framed Access.

Attribute Name	Command	
	ACR	ACA
Accounting-Auth-Method	0-1	0
Accounting-Input-Octets	1	0
Accounting-Output-Octets	1	0
Accounting-Record-Type	1	1
Accounting-Record-Number	0-1	0-1
Accounting-Realtime-Required	0-1	0-1
Accounting-Sub-Session-Id	0-1	0-1
Acct-Application-Id	0-1	0-1
Acct-Session-Id	1	0-1
Acct-Multi-Session-Id	0-1	0-1
Acct-Authentic	1	0
Acct-Delay-Time	0-1	0
Acct-Interim-Interval	0-1	0-1

Acct-Link-Count	0-1	0
Acct-Session-Time	1	0
Authorization-Lifetime	0-1	0
Callback-Id	0-1	0
Callback-Number	0-1	0
Called-Station-Id	0-1	0
Calling-Station-Id	0-1	0
Class	0+	0+
Connection-Info	0+	0
Destination-Host	0-1	0
Destination-Realm	1	0
Event-Timestamp	0-1	0-1
Error-Message	0	0-1
Error-Reporting-Host	0	0-1
Failed-AVP	0	0+
Login-IP-Host	0+	0
Login-IPv6-Host	0+	0
Login-LAT-Service	0-1	0
Login-LAT-Node	0-1	0
Login-LAT-Group	0-1	0
Login-LAT-Port	0-1	0
Login-Service	0-1	0
Login-TCP-Port	0-1	0
-----+-----+		

Attribute Name	Command	
	ACR	ACA
-----+-----+		
NAS-Identifier	0-1	0-1
NAS-IP-Address	0-1	0-1
NAS-IPv6-Address	0-1	0-1
NAS-Port	0-1	0-1
NAS-Port-Id	0-1	0-1
NAS-Port-Type	0-1	0-1
Origin-AAA-Protocol	0-1	0-1
Origin-Host	1	1
Origin-Realm	1	1
Origin-State-Id	0-1	0-1
Originating-Line-Info	0-1	0
Proxy-Info	0+	0+
QoS-Filter-Rule	0+	0
Route-Record	0+	0
Result-Code	0	1
Session-Id	1	1
Service-Type	0-1	0-1
Termination-Cause	0-1	0-1

User-Name	0-1 0-1
-----	-----+-----+

6. Unicode Considerations

A number of the AVPs in this RFC use the UTF8String type specified in the Diameter Base protocol [RFC6733]. Implementation differences in Unicode input processing may result in the same Unicode input characters generating different UTF-8 strings that fail to match when compared for equality. This may result in interoperability problems between a network access server and a Diameter server when a UTF-8 string entered locally is compared with one received via Diameter. Many of the uses of UTF8String in this RFC are limited to the 7-bit ASCII-compatible subset of UTF-8 where this class of Unicode string comparison problems does not arise.

Careful preparation of Unicode strings can increase the likelihood that string comparison will work in ways that make sense for typical users throughout the world; [RFC3454] is an example a framework for such Unicode string preparation. The Diameter application specified in this RFC has been deployed with use of Unicode in accordance with [RFC4005], which does not require any Unicode string preparation. As a result, additional requirements for Unicode string preparation in this RFC would not be backwards compatible with existing usage.

The Diameter server and the network access servers that it serves can be assumed to be under common administrative control, and all of the UTF-8 strings involved are part of the configuration of these servers. Therefore administrative interfaces for implementations of this RFC:

- a. SHOULD accept direct UTF-8 input of all configuration strings for AVPs that allow Unicode characters beyond the 7-bit ASCII-compatible subset of Unicode (in addition to any provisions for accepting Unicode characters for processing into UTF-8), and
- b. SHOULD make all such configuration strings available as UTF-8 strings

This functionality enables an administrator who encounters Unicode string comparison problems to copy one instance of a problematic UTF-8 string from one server to the other, after which the two (now identical) copies should compare as expected.

7. IANA Considerations

Several of the namespaces used in this document are managed by the Internet Assigned Numbers Authority [IANA], including the AVP Codes

[AVP-Codes], AVP Specific Values [AVP-Vals], Application IDs [App-Ids], Command Codes [Command-Codes] and RADIUS Attribute Values [RADIUSAttrVals].

For the current values allocated, and the policies governing allocation in those namespaces, please see the above-referenced registries.

IANA Note: Please change all the references in the registries listed above that are currently pointing to RFC 4005 to point to this document instead; please change the reference for the value '1' in the "Application IDs" sub-registry of the "Authentication, Authorization, and Accounting (AAA) Parameters" registry to point to this document, as well.

RFC Editor: Please remove both this note and the IANA note above before publication.

8. Security Considerations

This document describes the extension of Diameter for the NAS application. Security considerations regarding the Diameter protocol itself are discussed in [RFC6733]. Use of this application of Diameter MUST take into consideration the security issues and requirements of the Base protocol.

8.1. Authentication Considerations

This document does not contain a security protocol but does discuss how PPP authentication protocols can be carried within the Diameter protocol. The PPP authentication protocols described are PAP and CHAP.

The use of PAP SHOULD be discouraged, as it exposes users' passwords to possibly non-trusted entities. However, PAP is also frequently used for use with One-Time Passwords, which do not expose a security risk.

This document also describes how CHAP can be carried within the Diameter protocol, which is required for RADIUS backward compatibility. The CHAP protocol, as used in a RADIUS environment, facilitates authentication replay attacks.

The use of the EAP authentication protocols [RFC4072] can offer better security, given a method suitable for the circumstances.

Depending on the value of the Auth-Request-Type AVP, the Diameter protocol allows authorization-only requests that contain no

authentication information from the client. This capability goes beyond the Call Check capabilities provided by RADIUS (Section 5.6 of [RFC2865]) in that no access decision is requested. As a result, a new session cannot be started as a result of a response to an authorization-only request without introducing a significant security vulnerability.

8.2. AVP Considerations

Diameter AVPs often contain security-sensitive data; for example, user passwords and location data, network addresses and cryptographic keys. With the exception of the Configuration-Token (Section 4.4.8), QoS-Filter-Rule (Section 4.4.9) and Tunneling (Section 4.5.1) AVPs, all of the AVPs defined in this document are considered to be security-sensitive.

Diameter messages containing any AVPs considered to be security-sensitive MUST only be sent protected via mutually authenticated TLS or IPsec. In addition, those messages MUST NOT be sent via intermediate nodes unless there is end-to-end security between the originator and recipient or the originator has locally trusted configuration that indicates that end-to-end security is not needed. For example, end-to-end security may not be required in the case where an intermediary node is known to be operated as part of the same administrative domain as the endpoints so that an ability to successfully compromise the intermediary would imply a high probability of being able to compromise the endpoints as well. Note that no end-to-end security mechanism is specified in this document.

9. References

9.1. Normative References

- [ANITypes] NANPA Number Resource Info, "ANI Assignments", <http://www.nanpa.com/number_resource_info/ani_ii_assignments.html>.
- [RFC1994] Simpson, W., "PPP Challenge Handshake Authentication Protocol (CHAP)", RFC 1994, August 1996.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC2865] Rigney, C., Willens, S., Rubens, A., and W. Simpson, "Remote Authentication Dial In User Service (RADIUS)", RFC 2865, June 2000.

- [RFC3162] Aboba, B., Zorn, G., and D. Mitton, "RADIUS and IPv6", RFC 3162, August 2001.
- [RFC3516] Nerenberg, L., "IMAP4 Binary Content Extension", RFC 3516, April 2003.
- [RFC3539] Aboba, B. and J. Wood, "Authentication, Authorization and Accounting (AAA) Transport Profile", RFC 3539, June 2003.
- [RFC5777] Korhonen, J., Tschofenig, H., Arumaithurai, M., Jones, M., and A. Lior, "Traffic Classification and Quality of Service (QoS) Attributes for Diameter", RFC 5777, February 2010.
- [RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.

9.2. Informative References

- [ARAP] Apple Computer, "Apple Remote Access Protocol (ARAP) Version 2.0 External Reference Specification", R0612LL/B , September 1994.
- [AVP-Codes] IANA, "IANA AAA AVP Codes Registry", <<http://www.iana.org/assignments/aaa-parameters/aaa-parameters.xml#aaa-parameters-1>>.
- [AVP-Vals] IANA, "IANA AAA AVP Specific Values", <<http://www.iana.org/assignments/aaa-parameters/aaa-parameters.xml#aaa-parameters-2>>.
- [App-Ids] IANA, "IANA AAA Application IDs Registry", <<http://www.iana.org/assignments/aaa-parameters/aaa-parameters.xml#aaa-parameters-1>>.
- [AppleTalk] Sidhu, G., Andrews, R., and A. Oppenheimer, "Inside AppleTalk", Second Edition Apple Computer, 1990.
- [BASE] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.
- [Command-Codes] IANA, "IANA AAA Command Codes Registry", <<http://www.iana.org/assignments/aaa-parameters/aaa-parameters.xml#command-code-rules>>.

- [IANA] IANA, "Internet Assigned Numbers Authority", <<http://www.iana.org/>>.
- [IPX] Novell, Inc., "NetWare System Technical Interface Overview", #883-000780-001, June 1989.
- [ISO.8859-1.1987] International Organization for Standardization, "Information technology - 8-bit single byte coded graphic - character sets - Part 1: Latin alphabet No. 1, JTC1/SC2", ISO Standard 8859-1, 1987.
- [LAT] Digital Equipment Corp., "Local Area Transport (LAT) Specification V5.0", AA-NL26A-TE, June 1989.
- [RADIUSAttrVals] IANA, "IANA Radius Attribute Values Registry", <<http://www.iana.org/assignments/radius-types/radius-types.xml#radius-types-3>>.
- [RFC1334] Lloyd, B. and W. Simpson, "PPP Authentication Protocols", RFC 1334, October 1992.
- [RFC1661] Simpson, W., "The Point-to-Point Protocol (PPP)", STD 51, RFC 1661, July 1994.
- [RFC1990] Sklower, K., Lloyd, B., McGregor, G., Carr, D., and T. Coradetti, "The PPP Multilink Protocol (MP)", RFC 1990, August 1996.
- [RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.
- [RFC2548] Zorn, G., "Microsoft Vendor-specific RADIUS Attributes", RFC 2548, March 1999.
- [RFC2597] Heinanen, J., Baker, F., Weiss, W., and J. Wroclawski, "Assured Forwarding PHB Group", RFC 2597, June 1999.
- [RFC2637] Hamzeh, K., Pall, G., Verthein, W., Taarud, J., Little, W., and G. Zorn, "Point-to-Point Tunneling Protocol", RFC 2637, July 1999.
- [RFC2866] Rigney, C., "RADIUS Accounting", RFC 2866, June 2000.

- [RFC2867] Zorn, G., Aboba, B., and D. Mitton, "RADIUS Accounting Modifications for Tunnel Protocol Support", RFC 2867, June 2000.
- [RFC2868] Zorn, G., Leifer, D., Rubens, A., Shriver, J., Holdrege, M., and I. Goyret, "RADIUS Attributes for Tunnel Protocol Support", RFC 2868, June 2000.
- [RFC2869] Rigney, C., Willats, W., and P. Calhoun, "RADIUS Extensions", RFC 2869, June 2000.
- [RFC2881] Mitton, D. and M. Beadles, "Network Access Server Requirements Next Generation (NASREQNG) NAS Model", RFC 2881, July 2000.
- [RFC2989] Aboba, B., Calhoun, P., Glass, S., Hiller, T., McCann, P., Shiino, H., Walsh, P., Zorn, G., Dommety, G., Perkins, C., Patil, B., Mitton, D., Manning, S., Beadles, M., Chen, X., Sivalingham, S., Hameed, A., Munson, M., Jacobs, S., Lim, B., Hirschman, B., Hsu, R., Koo, H., Lipford, M., Campbell, E., Xu, Y., Baba, S., and E. Jaques, "Criteria for Evaluating AAA Protocols for Network Access", RFC 2989, November 2000.
- [RFC3169] Beadles, M. and D. Mitton, "Criteria for Evaluating Network Access Server Protocols", RFC 3169, September 2001.
- [RFC3246] Davie, B., Charny, A., Bennet, J., Benson, K., Le Boudec, J., Courtney, W., Davari, S., Firoiu, V., and D. Stiliadis, "An Expedited Forwarding PHB (Per-Hop Behavior)", RFC 3246, March 2002.
- [RFC3454] , .
- [RFC3580] Congdon, P., Aboba, B., Smith, A., Zorn, G., and J. Roese, "IEEE 802.1X Remote Authentication Dial In User Service (RADIUS) Usage Guidelines", RFC 3580, September 2003.
- [RFC3931] Lau, J., Townsley, M., and I. Goyret, "Layer Two Tunneling Protocol - Version 3 (L2TPv3)", RFC 3931, March 2005.
- [RFC4072] Eronen, P., Hiller, T., and G. Zorn, "Diameter Extensible Authentication Protocol (EAP) Application", RFC 4072, August 2005.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

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

Appendix A. Acknowledgements

A.1. This Document

The vast majority of the text in this document was taken directly from RFC 4005; the editor owes a debt of gratitude to the authors thereof (especially Dave Mitton, who somehow managed to make nroff paginate the AVP Occurance Tables correctly!).

Thanks (in no particular order) to Jai-Jin Lim, Liu Hans, Sebastien Decugis, Jouni Korhonen, Mark Jones, Hannes Tschofenig, Dave Crocker, David Black, Barry Leiba, Peter Saint-Andre, Stefan Winter and Lionel Morand for their useful reviews and helpful comments.

A.2. RFC 4005

The authors would like to thank Carl Rigney, Allan C. Rubens, William Allen Simpson, and Steve Willens for their work on the original RADIUS protocol, from which many of the concepts in this specification were derived. Thanks, also, to Carl Rigney for [RFC2866] and [RFC2869]; Ward Willats for [RFC2869]; Glen Zorn, Bernard Aboba, and Dave Mitton for [RFC2867] and [RFC3162]; and Dory Leifer, John Shriver, Matt Holdrege, Allan Rubens, Glen Zorn and Ignacio Goyret for their work on [RFC2868]. This document stole text and concepts from both [RFC2868] and [RFC2869]. Thanks go to Carl Williams for providing IPv6-specific text.

The authors would also like to acknowledge the following people for their contributions in the development of the Diameter protocol: Bernard Aboba, Jari Arkko, William Bulley, Kuntal Chowdhury, Daniel C. Fox, Lol Grant, Nancy Greene, Jeff Hagg, Peter Heitman, Paul Krumviede, Fergal Ladley, Ryan Moats, Victor Muslin, Kenneth Peirce, Sumit Vakil, John R. Vollbrecht, and Jeff Weisberg.

Finally, Pat Calhoun would like to thank Sun Microsystems, as most of the effort put into this document was done while he was in their employ.

Author's Address

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

Phone: +66 (0)8-1000-4155
EMail: glenzorn@gmail.com

Diameter Maintenance and Extensions
(DIME)
Internet-Draft
Intended status: Standards Track
Expires: August 29, 2013

J. Korhonen
Renesas Mobile
H. Tschofenig, Ed.
Nokia Siemens Networks
February 25, 2013

The Diameter Overload Control Application (DOCA)
draft-korhonen-dime-ovl-01.txt

Abstract

This specification documents a Diameter Overload Control Application (DOCA), which uses the normal Diameter application approach for the capability negotiation, propagation and management of Diameter overload control information between Diameter nodes.

Status of this Memo

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

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

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

This Internet-Draft will expire on August 29, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Requirements	3
3. DOCA Overview	4
4. DOCA Commands	5
5. Attribute Value Pairs	6
5.1. OC-Information AVP	6
5.2. OC-Scope AVP	7
5.3. OC-Applications AVP	8
5.4. OC-Action AVP	9
5.5. OC-Algorithm AVP	9
5.6. OC-Level AVP	10
5.7. OC-Utilization AVP	11
5.8. OC-Tocl AVP	11
5.9. OC-Sending-Rate AVP	11
5.10. OC-Best-Before AVP	12
5.11. OC-Origin AVP	12
5.12. OC-Priority AVP	12
5.13. Attribute Value Pair flag rules	13
6. Transport Considerations	13
7. Examples	14
8. IANA Considerations	15
8.1. Application Identifiers	15
8.2. SCTP Payload Protocol Identifier	15
8.3. Command Codes	15
8.4. AVP Codes	15
8.5. Result-Code Values	15
8.6. New Registries	16
9. Security Considerations	16
10. Acknowledgements	16
11. References	17
11.1. Normative References	17
11.2. Informative References	17
Appendix A. Design Justification	17
Authors' Addresses	18

1. Introduction

The existing toolbox offered by the Diameter Base Protocol [RFC6733] to prevent and recover from signaling overload situations is rather limited. Apart from out-of-band altering of the transport connection congestion control behavior or other non-standard application level throttling, the protocol error DIAMETER_TOO_BUSY, the permanent error DIAMETER_UNABLE_TO_COMPLY (for some unspecified reason) and the Disconnect-Cause Attribute Value Pair (AVP) code BUSY or DO_NOT_WANT_TO_TALK_TO_YOU are more or less all there is. Unfortunately, the mentioned three indications are coarse, concern one peer connection at a time or lack detailed information for problem diagnosis and mitigation. They also treat all applications in a single Diameter node (identified by a single DiameterIdentity) as a lump. There is no way communicate any kind of grouping of applications or what is the scope/partitioning of the delivered information. Furthermore, there is no way to signal when the overload situation is over. The request initiator and forwarders are therefore forced to keep re-submitting their messages to determine whether the situation has changed.

The situation is further complicated by the hop-by-hop nature of Diameter deployments. This makes the propagation of possible overload situation information non-trivial, even for existing protocol errors since every intermediate Diameter node is allowed to react to the error situation. Either the information is never propagated to the originator of the request or it takes an unacceptable long time.

A problem statement of overload control for Diameter and requirements are documented in [I-D.ietf-dime-overload-reqs]. This specification describes a solution to the Diameter overload Diameter Overload Control Application (DOCA), which fulfills the requirements of [I-D.ietf-dime-overload-reqs] and defines a Diameter application to convey overload information between Diameter nodes.

Note: The recent publication of [I-D.campbell-dime-overload-data-analysis] illustrates the overload information data model and the design space. As the working group makes progress in deciding about specific features this document will be updated accordingly.

2. Requirements

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

3. DOCA Overview

Any the DOCA capable Diameter node MAY initiate a DOCA-Report-Request at any given time. The receiver of the DOCA-Report-Request acknowledges with a DOCA-Report-Answer and includes the Result-Code AVP indicating whether it could honor the action/report in the request. The DOCA-Report-Answer SHOULD also piggyback overload control information.

A DOCA client MUST set the Auth-Session-State AVP to the value NO_STATE_MAINTAINED and SHOULD include the OC-Information AVP with overload information into the DOCA-Report-Request, if available. The DOCA-Report-Response message MUST contain the Auth-Session-State AVP set to value NO_STATE_MAINTAINED.

Note that information exchanges regarding various DOCA related timers serve only as a hint since they cannot be enforced. Consequently, care should be taken not to send DOCA-Report-Requests too frequently.

When a Diameter node receives overload control information and is also requested to act on it, the DOCA functionality is applied to all specified applications within a given scope. How the Diameter node accomplishes the node wide DOCA action enforcement is implementation specific.

When a Diameter node receives (interim) overload information but the overload condition has not exceeded a certain threshold, then the receiver is not required to act based on the received information. However, it is RECOMMENDED that the receiver makes proactive actions to avoid entering the overload condition based on the newly received overload information.

There may be zero or more intermediate Diameter agents on the path between the DOCA client and the DOCA server. Understanding the DOCA functionality is not expected from relays and redirect agents. A Diameter proxy, which obviously understands the DOCA application, MAY inspect the DOCA related AVPs in the DOCA-Report-Request/Answer message pair and depending on the value of the OC-Scope AVP (see Section 5.2) inject its own information. A proxy is always RECOMMENDED to react according to the overload information when it comes to, for example, peer selection and traffic throttling.

When a Diameter agent receives overload control information and is also requested to act on it, the DOCA functionality is applied to all specified applications within a given scope. How the Diameter agent accomplishes the node wide DOCA action enforcement is implementation specific.

4. DOCA Commands

The DOCA-Report-Request (DRR) message is used to report overload condition information. The message can be originated as a result of emerging overload condition or as a periodic unsolicited report.

```
<DOCA-Report-Request> ::= < Diameter Header: TBD2, REQ, PXY >
    < Session-Id >
    { Auth-Application-Id }
    { Origin-Host }
    { Origin-Realm }
    { Destination-Realm }
    { Auth-Request-Type }
    { Destination-Host }
    [ Auth-Session-State ]
    * [ Class ]
    [ Origin-State-Id ]
    * [ Proxy-Info ]
    * [ Route-Record ]

    { OC-Scope }
    [ OC-Algorithm ]
    [ OC-Action ]
    [ OC-Tocl ]
    [ OC-Applications ]
    * [ OC-Information ]

    * [ AVP ]
```

The DOCA-Report-Answer (DRA) message is used as a response to the DOCA-Report-Request. The message MAY piggyback overload condition information in order to avoid unnecessary DOCA-Report-Request messages to the reverse direction.

```
<DOCA-Report-Answer> ::= < Diameter Header: TBD2, PXY >
    < Session-Id >
    { Result-Code }
    { Origin-Host }
    { Origin-Realm }
    [ Auth-Session-State ]
    * [ Class ]
    [ Error-Message ]
    [ Error-Reporting-Host ]
    [ Failed-AVP ]
    [ Origin-State-Id ]
    * [ Redirect-Host ]
    [ Redirect-Host-Usage ]
    [ Redirect-Max-Cache-Time ]
    * [ Proxy-Info ]

    { OC-Scope }
    [ OC-Action ]
    * [ OC-Information ]

    * [ AVP ]
```

5. Attribute Value Pairs

5.1. OC-Information AVP

The OC-Information AVP (AVP Code TBD3) is of type Grouped and contains a set AVPs that identify the source of the overload control information (the OC-Origin AVP), the overload information itself and which applications the information concerns.

```
OC-Information ::= < AVP Header: TBD3 >
    { OC-Origin }
    { OC-Best-Before }
    [ OC-Level ]
    [ OC-Algorithm ]
    [ OC-Sending-Rate ]
    [ Vendor-Id ]
    [ OC-Applications ]
    [ Product-Name ]
    [ OC-Utilization ]
    [ OC-Priority ]
    * [ AVP ]
```

Diameter proxies on path MAY add one or more OC-Information AVPs into the DOCA-Report-Request/answer messages.

5.2. OC-Scope AVP

The OC-Scope (AVP Code TBD4) is of type Unsigned32 and contains the scope where and concerning what the overload control information can be injected. The OC-Scope is formatted as a vector of scope flag bits. The following scopes are supported:

Host scope (0x00000001)

The OC-Information AVP concerns only a single host within a realm (which internally MAY represent of pool).

Realm scope (0x00000002)

The OC-Information AVP concerns a realm. No specific hosts are identified.

Only origin realm (0x00000004)

The OC-Information AVP can only be included by a Diameter node on the path that has the same Origin-Realm as the DOCA client.

Application information (0x00010000)

The OC-Information AVP MAY contain application related information (the OC-Applications AVP).

Node utilization information (0x00020000)

The OC-Information AVP MAY contain node wide load related information (the OC-Utilization AVP).

Application priorities (0x00040000)

The OC-Information AVP SHOULD priority information (the OC-Priority AVP) so when the overload condition is on, Diameter nodes are able to prioritize between different applications, for example, when dropping or throttling messages.

Any other value is reserved.

A scope is active when a corresponding flag is set in the OC-Scope AVP. During the initialization state a DOCA client includes those scopes it supports and is interested in. A DOCA server then returns the scope that it has in common with the DOCA client (and intends to use). The common scopes are then used during the established state. Note that some scope combinations make little sense while still being valid. The general guide when multiple scopes collide is that the

least restrictive wins.

A sender of the overload information MUST adhere to the scope it announces regarding the information it itself sends.

If a DOCA server does not have a common scope with a DOCA client or the DOCA server cannot agree on one based on a local policy, then the DOCA server MUST send the DOCA-Report-Answer indicating an error and set the Result-Code to the DIAMETER_NO_COMMON_SCOPE value.

5.3. OC-Applications AVP

The OC-Applications (AVP Code TBD5) is of type Grouped and contains a list of Application-IDs of interest when found in the DOCA-Report-Request/Answer command main level and meant to be used during the initialization state to agree on the common set of supported applications of monitoring interest. When used within the OC-Information AVP, the OC-Applications AVP identify those applications the overload information concerns. The OC-Applications AVPs on the command main level and inside the OC-Information AVP MUST NOT have conflicting views of the applications of interest. However, the OC-Applications AVP can be seen as a superset of applications i.e., not all applications of interest need to be included every time into the OC-Information AVP.

```
OC-Applications ::= < AVP Header: TBD3 >
                  * [ Auth-Application-Id ]
                  * [ Acct-Application-Id ]
                  * [ Vendor-Specific-Application-Id ]
                  * [ AVP ]
```

The absence of the OC-Applications AVP indicates the Diameter node has no specific preference or interest in specific applications. The overload information is then signaled as concerning the whole Diameter node. This default behavior is useful when the DOCA does not maintain session state. If there are no common applications, then the DOCA-Report-Answer MUST contain the Result-Code with the DIAMETER_NO_COMMON_APPLICATION value.

When the DOCA maintains state, there is no need to include the OC-Applications AVP into the DOCA-Report-Request/Answer command main level after the initial message exchange. The agreed common set of application is expected to be known by both DOCA client and server throughout the session lifetime.

5.4. OC-Action AVP

The OC-Action (AVP Code TBD6) is of type OctetString and size of one octet. The octet has the following three possible values:

Start (1)

Signals the start of the overload condition. This implies the receiver is requested to act according to the information found in the OC-Information.

Stop (2)

Signals the end of the overload condition.

Interim (3)

Updates the overload information. The interim can be sent during the overload condition or during the normal condition. This is the default value.

Any other value is reserved.

5.5. OC-Algorithm AVP

The OC-Algorithm (AVP Code TBD7) is of type Unsigned32. The contains supported 'algorithms' to mitigate the overload condition. The OC-Algorithm AVP is formatted as a vector of algorithm flag bits. The following 'algorithms' are supported:

Drop (0x00000001)

Messages are plain dropped. It is RECOMMENDED to drop messages selectively based, for example, on application priorities. This is the default algorithm.

Throttle (0x00000002)

The message sending rate is according to the OC-Sending-Rate AVP.

Prioritize (0x00000004)

Apply priorities among applications and the other used means for holding traffic.

Any other value is reserved.

The 'algorithms' are only applied at a Diameter node when the

overload condition has been signaled.

During the initialization state a DOCA client includes those algorithms it supports and is interested in. A DOCA server then returns the algorithm that it has in common with the DOCA client (and intends to use). One or more common algorithms are then used during the established state.

If a DOCA server does not have a common algorithm with a DOCA client or the DOCA server cannot agree on one based on a local policy, then the DOCA server **MUST** send the DOCA-Report-Answer indicating an error and set the Result-Code to the `DIAMETER_NO_COMMON_ALGORITHM` value.

5.6. OC-Level AVP

The OC-Level (AVP Code TBD8) is of type OctetString and size of one octet. The octet has the following five possible values:

Normal (1)

Everything is in control. Meaningful only when the OC-Action is set to 'Interim' since when the overload condition level is considered normal, the overload condition **SHOULD** be stopped. This is the default value.

Raising (2)

There is a sign of increasing load.

Alarming (3)

The overload condition is reaching the level where quick measures **SHOULD** be done to mitigate the overload condition.

Panic (4)

The overload condition is severe. Apply any measure to mitigate the overload condition but still allowed to send messages.

Hold (5)

Do not send any messages, please. When this level is signaled, the OC-Best-Before time **SHOULD NOT** be respected but an explicit overload condition stop has to be received (with an exception the Diameter node realizes its other end has rebooted or otherwise lost its state).

Switch servers (6)

Do not talk to me again. When this level is signaled, the DOCA client MUST switch to an alternative server.

Any other value is reserved.

If the receiver cannot agree on or does not understand the OC-Level AVP value, the an error MUST be returned with the Result-Code AVP set to the value DIAMETER_INVALID_AVP_VALUE and the Failed-AVP AVP containing the OC-Level AVP.

5.7. OC-Utilization AVP

The OC-Utilization (AVP Code TBD9) is of type Float32 and tells the overall utilization level percentage of the Diameter node. Values between 0.0 to 100.0 are valid.

5.8. OC-Tocl AVP

The OC-Tocl (AVP Code TBD10) is of type Unsigned32 and tells the Tocl timer value in milliseconds. This timer defines the interval for sending periodic DOCA-Report-Request messages with the OC-Action AVP set to 'Interim'. The value of zero (0) means no periodic DOCA-Report-Request messages are sent or desired. The default value is 120000.

The OC-Tocl AVP can be considered as a hint for a desired sending rate of subsequent messages.

If a DOCA server find the Tocl value proposed by a DOCA client either too small (i.e. too frequent periodic messages) or too big (i.e. too seldom periodic messages), then the DOCA server MUST send the DOCA-Report-Answer indicating an error and set the Result-Code either to the DIAMETER_TOCL_TOO_SMALL or DIAMETER_TOCL_TOO_BIG value.

5.9. OC-Sending-Rate AVP

The OC-Sending-Rate (AVP Code TBD11) is of type Float32 and tells the the maximum Diameter message sending rate per second the sender of this information wishes to receive Diameter messages. Only positive values are valid. A value of zero (0.0) of the absence of this AVP means the information sender has no specific rate preference.

If a DOCA server finds the sending rate value proposed by a DOCA client too big (i.e., too frequent periodic messages), then the DOCA server MUST send the DOCA-Report-Answer indicating an error and set

the Result-Code to the DIAMETER_RATE_TOO_BIG value.

5.10. OC-Best-Before AVP

The OC-Best-Before (AVP Code TBD12) is of type Time and tells the expiration time/date for the information received in the OC-Information. For example, when the overload condition is on, the expiration of the 'best before' timer causes the same as receiving a DOCA-Report-Request/Answer with the OC-Action set to 'Stop'.

[Editor's note: to be decided whether a duration timer is a better measure. Using Time has the assumptions nodes have actually clocks that a running approximately same time.]

5.11. OC-Origin AVP

The OC-Origin (AVP Code TBD13) is of type DiameterIdentity and tells the identity of the Diameter node that originated included the overload control information. Both host and realm information MUST be included in the OC-Origin AVP. Note, if the OC-Scope AVP indicates only a realm wide scope for the overload information, then the realm part of the OC-Origin AVP is meaningful and the host information only serves as an additional information of the representative for the realm wide information.

5.12. OC-Priority AVP

The OC-Priority (AVP Code TBD14) is of type Unsigned32 and defines the priority level. The value of 0x00000000 is the highest priority and the value of 0xffffffff is the lowest priority. The absence of the OC-Priority AVP means there is not specific priority level defined and the priority SHOULD be considered as the lowest possible.

When used within the OC-Information grouped AVP, the OC-Priority AVP defines the priority for the listed applications within the OC-Applications AVP.

5.13. Attribute Value Pair flag rules

				+-----+ AVP flag rules +-----+	
Attribute Name	AVP Code	Section Defined	Value Type		MUST
				MUST	NOT
OC-Information	TBD3	x.x	Grouped	M	V
OC-Scope	TBD4	x.x	Unsigned32	M	V
OC-Application	TBD5	x.x	Grouped	M	V
OC-Action	TBD6	x.x	OctetString	M	V
OC-Algorithm	TBD7	x.x	Unsigned32	M	V
OC-Level	TBD8	x.x	OctetString	M	V
OC-Utilization	TBD9	x.x	Float32	M	V
OC-Tocl	TBD10	x.x	Unsigned32	M	V
OC-Sending-Rate	TBD11	x.x	Float32	M	V
OC-Best-Before	TBD12	x.x	Time	M	V
OC-Origin	TBD13	x.x	DiameterIdentity	M	V
OC-Priority	TBD14	x.x	Unsigned32	M	V

6. Transport Considerations

In case of Stream Control Transmission Protocol (SCTP) transport, the DOCA application is RECOMMENDED to mark its Diameter packets using the DOCA defined SCTP Payload Protocol Identifier (PPID) TBD1. The PPID MAY be used by intermediating network nodes or agents to peek into SCTP message and find out that this is about overload control. Such information can be used for prioritizing SCTP packet handling as an example.

7. Examples

Consider the following simplified scenario shown in Figure 1 where two servers are connected to a proxy. All three nodes understand the DOCA application. These three nodes belong to the same administrative domain and the operator decided that he wants to hide the Diameter topology of his own network. Consequently, aggregate information is provided by the proxy for any Diameter overload message exchange. The Diameter client also supports the DOCA application. Between the client and the Diameter proxy we assume an arbitrary Diameter network that passes Diameter messages back and forth.

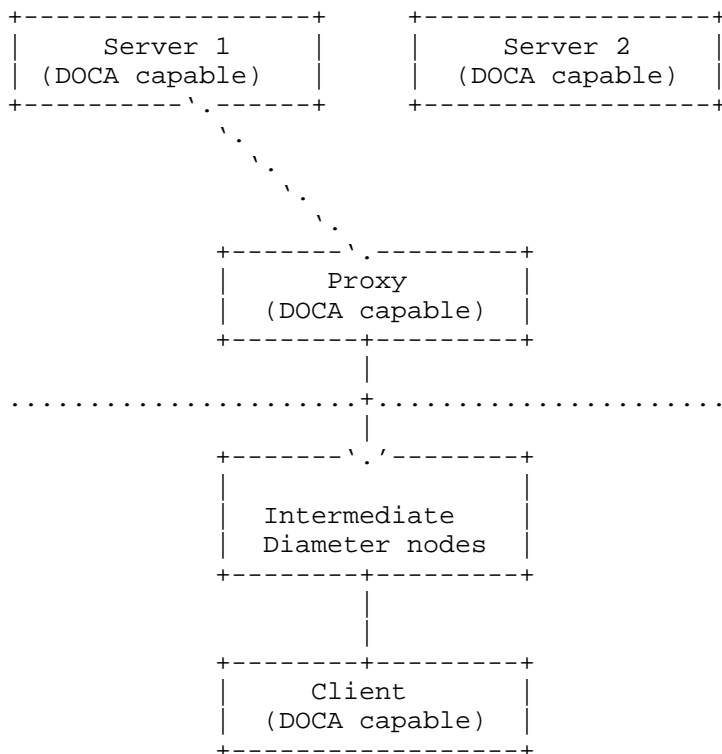


Figure 1

Let us assume that the DOCA exchange is initiated by server 1 who determines that the load situation increases. It sends a DOCA-Report-Request message (with piggybacked overload information) towards the client. The message also instructs the client to reduce

it's sending rate. The proxy, who receives the DOCA-Report-Request decides to change the included OC-Origin information and forwards the request to the client.

When the client receives the DOCA-Report-Request message is processes the content, evaluates the overload information content and reacts accordingly, and returns a DOCA-Report-Answer message back to acknowledge the receipt.

Alternatively, let us assume that the proxy does not forward the message but instead terminates the DOCA-Report-Request received from Server 1. It instead decides to route traffic to the backup server, Server 2. In this case the entire process was transparent for the client.

8. IANA Considerations

8.1. Application Identifiers

This specification reserves a new Diameter Application-ID TBD14 for the Diameter Overload Control Application (DOCA) from the 'Authentication, Authorization, and Accounting (AAA) Parameters' Application IDs registry.

8.2. SCTP Payload Protocol Identifier

Section 6 reserves a new SCTP Payload Protocol Identifier for the DOCA application usage. The value is reserved from the existing SCTP Payload Protocol Identifiers registry.

8.3. Command Codes

Two command codes are defined in Section 4. The DOCA-Report-Request Command Code is TBD and the DOCA-Report-Answer Command Code is TBD. Both are allocated from the 'Authentication, Authorization, and Accounting (AAA) Parameters' Command Codes registry.

8.4. AVP Codes

New AVPs defined by this specification are listed in Section 5. All AVP codes allocated from the 'Authentication, Authorization, and Accounting (AAA) Parameters' AVP Codes registry.

8.5. Result-Code Values

This specification adds several Diameter Overload Control Application specific Permanent Failure codes from the 'Authentication,

Authorization, and Accounting (AAA) Parameters' Result-Code AVP Values (code 268) - Permanent Failure registry:

AVP Values	Attribute Name	Reference
5xxx	DIAMETER_NO_COMMON_SCOPE	RFCxxxx
5xxx	DIAMETER_NO_COMMON_ALGORITHM	RFCxxxx
5xxx	DIAMETER_TOCL_TOO_SMALL	RFCxxxx
5xxx	DIAMETER_TOCL_TOO_BIG	RFCxxxx
5xxx	DIAMETER_RATE_TOO_BIG	RFCxxxx

8.6. New Registries

Four new registries are needed under the 'Authentication, Authorization, and Accounting (AAA) Parameters' registry:

- o OC-Scope AVP Values: the policy for this registry is Specification Required.
- o OC-Action AVP Values: the policy for this registry is Standards Action.
- o OC-Level AVP Values: the policy for this registry is Standards Action.
- o OC-Algorithm AVP Values: the policy for this registry is Specification Required.

9. Security Considerations

The security properties of the Diameter Overload Control Application (DOCA) follows the security model of Diameter [RFC6733]. This implies there is no proper means to verify the message and AVP content correctness if multiple intermediate Diameter agents are present on the path between the DOCA client and server. As a result a malicious intermediate could feed incorrect overload control information to DOCA clients and peers, and thus affect negatively to the overload condition recovery. A possible way to overcome the obvious security vulnerability is to mandate the use of end-to-end security at the Diameter AVP level.

As such, like any other Diameter application this document would benefit from a Diameter end-to-end security mechanism. While work is in progress it has not yet been finalized and therefore this specification does not rely on it.

10. Acknowledgements

The author thanks Annett Seefeldt for her constructive comments on

the technical aspects on this document.

11. References

11.1. Normative References

- [I-D.ietf-dime-overload-reqs]
McMurry, E. and B. Campbell, "Diameter Overload Control Requirements", draft-ietf-dime-overload-reqs-04 (work in progress), February 2013.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.

11.2. Informative References

- [I-D.campbell-dime-overload-data-analysis]
Campbell, B., Tschofenig, H., Korhonen, J., and A. Roach, "Diameter Overload Data Analysis", draft-campbell-dime-overload-data-analysis-00 (work in progress), February 2013.
- [RFC6408] Jones, M., Korhonen, J., and L. Morand, "Diameter Straightforward-Naming Authority Pointer (S-NAPTR) Usage", RFC 6408, November 2011.

Appendix A. Design Justification

Section 1 discussed the motivation and the background for the Diameter enhancements for an explicit Diameter overload control solution. This specification solves the overload control at the application level instead of extending the Diameter base protocol or piggybacking overload control information on top of existing applications. The reasoning is the following:

1. The support for Diameter overload control capability between Diameter peers is explicit (i.e., a new application-id is advertised) and thus not build on an exchange of optional Attribute Value Pairs (AVPs).
2. The support for Diameter overload control capability between Diameter client and server is explicit.

3. The peer selection follows the existing standards including DNS-based discovery [RFC6408] and does not assume additional peer selection criteria learnt from an exchange of optional AVPs.
4. The application based solution is able to traverse and also propagate overload control information through realms that deploy relay agents without Diameter overload control support.
5. The propagation does not depend on a modified behavior of already specified Diameter command codes.
6. Pretending not to establish a state when there actually is an overload capability and information state still maintained. The state might not be at the application level but is there.
7. Trying to avoid information flooding, especially across administrative domains.
8. The use of the application concept allows established mechanisms for filtering and Diameter traffic engineering, since it behaves like any other Diameter application.
9. The use of the dedicated application allows to isolate (even physically) the overload signaling into a dedicated transport that is not affected by other Diameter messages and network traffic.

Authors' Addresses

Jouni Korhonen
Renesas Mobile
Porkkalankatu 24
Helsinki 00180
Finland

Email: jouni.nospam@gmail.com

Hannes Tschofenig (editor)
Nokia Siemens Networks
Linnoitustie 6
Espoo 02600
Finland

Phone: +358 (50) 4871445
Email: Hannes.Tschofenig@gmx.net
URI: <http://www.tschofenig.priv.at>

DIME
Internet-Draft
Intended status: Standards Track
Expires: November 18, 2013

A. B. Roach
Mozilla
E. McMurry
Tekelec
May 17, 2013

A Mechanism for Diameter Overload Control
draft-roach-dime-overload-ctrl-03

Abstract

When a Diameter server or agent becomes overloaded, it needs to be able to gracefully reduce its load, typically by informing clients to reduce or stop sending traffic for some period of time. Otherwise, it must continue to expend resources parsing and responding to Diameter messages.

This document proposes a concrete, application-independent mechanism to address the challenge of communicating load and overload state among Diameter peers, and specifies an algorithm for load abatement to address such overload conditions as they occur. The load abatement algorithm is extensible, allowing for future documents to define additional load abatement approaches.

Status of this Memo

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

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

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

This Internet-Draft will expire on November 18, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal

Provisions Relating to IETF Documents
(<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	4
1.1. Mechanism Properties	4
1.2. Overview of Operation	6
1.3. Documentation Conventions	6
2. Overload Scopes	6
2.1. Scope Descriptions	7
2.2. Combining Scopes	8
3. Diameter Node Behavior	9
3.1. Connection Establishment Procedures	9
3.2. Diameter Client and Diameter Server Behavior	11
3.2.1. Sending a Request to a Compliant Peer	12
3.2.2. Receiving a Request	13
3.2.3. Sending an Answer to a Compliant Peer	14
3.2.4. Receiving an Answer from a Compliant Peer	15
3.3. Diameter Agent Behavior	16
3.3.1. Proxying a Request	16
3.3.2. Proxying an Answer	16
3.4. Proactive Load and Overload Communication	17
3.5. Load Processing	17
3.5.1. Sending Load Information	17
3.5.2. Receiving Load Information	18
3.6. Session Establishment for Session Groups	20
3.6.1. Session Group Concepts	20
3.6.2. Session Group Procedures	22
4. Loss-Based Overload Control Algorithm	22
4.1. Overload-Metric values for the 'Loss' Algorithm	23
4.2. Example Implementation	24
5. Diameter AVPs for Overload	28
5.1. Load-Info AVP	28
5.2. Supported-Scopes AVP	28
5.3. Overload-Algorithm AVP	29
5.4. Overload-Info-Scope AVP	30
5.4.1. Realm Scope	31
5.4.2. Application-ID Scope	31
5.4.3. Host Scope	31
5.4.4. Session Scope	31

5.4.5.	Connection Scope	31
5.4.6.	Session Group Scope	32
5.5.	Overload-Metric AVP	32
5.6.	Period-Of-Validity AVP	32
5.7.	Session-Group AVP	32
5.8.	Load AVP	32
6.	Security Considerations	32
7.	IANA Considerations	33
7.1.	New Diameter AVPs	33
7.2.	New Diameter Disconnect-Cause	33
7.3.	New Diameter Response Code	34
7.4.	New Command Flag	34
7.5.	Overload Algorithm Registry	34
7.6.	Overload Scope Registry	34
8.	References	35
8.1.	Normative References	35
8.2.	Informative References	35
Appendix A.	Acknowledgements	35
Appendix B.	Requirements Analysis	36
Appendix C.	Extending the Overload Mechanism	45
C.1.	New Algorithms	45
C.2.	New Scopes	46
Appendix D.	Design Rationale	46
D.1.	Piggybacking	47
D.2.	Load AVP in All Packets	48
D.3.	Graceful Failure	48
Authors' Addresses	49

1. Introduction

When a Diameter [RFC6733] server or agent becomes overloaded, it needs to be able to gracefully reduce its load, typically by informing clients to reduce or stop sending traffic for some period of time. Otherwise, it must continue to expend resources parsing and responding to Diameter messages.

This document defines a mechanism for communicating the load and overload information among Diameter nodes. It also defines a base algorithm for shedding traffic under overload circumstances. The design of the mechanism described in this document allows for the definition of alternate load abatement algorithms as well.

The mechanism proposed in this document is heavily influenced by the work performed in the IETF Session Initiation Protocol (SIP) Overload Control Working Group, and draws on the conclusions reached by that working group after extensive network modeling.

The solution described in this document is intended to satisfy the requirements described in [I-D.ietf-dime-overload-reqs], with the exception of REQ 34. As discussed in that document, the intention of a Diameter overload mechanism is to handle overload of the actual message processing portions of Diameter servers. This is in contrast to congestion, which is the inability of the underlying switching and routing fabric of the network to carry the volume of traffic at the volume that IP hosts wish to send it. Handling of congestion is relegated to the underlying transport protocol (TCP or SCTP), and will not be discussed.

Philosophically, the approach in designing this mechanism is based on the prospect that building a base-level, fully compliant implementation should be a very simple and straightforward exercise. However, the protocol includes many additional features that may be implemented to allow Diameter nodes to apply increasingly sophisticated behaviors. This approach gives implementors the freedom to implement as sophisticated a scheme as they desire, while freeing them from the burden of unnecessary complexity. By doing so, the mechanism allows for the rapid development and deployment of the mechanism followed by a period of steady and gradual improvements as implementations become more capable.

1.1. Mechanism Properties

The core Diameter overload mechanism described in this document is fundamentally hop-by-hop. The rationale for using a hop-by-hop approach is the same as is described in section 5.1 of [RFC6357]. However, due to the fact that Diameter networks frequently have

traffic that is easily grouped into a few well-defined categories, we have added some concepts that allow Diameter agents to push back on subsets of traffic that correspond to certain well-defined and client-visible constructs (such as Destination-Host, Destination-Realm, and Application-ID). These constructs are termed "Scopes" in this document. A more complete discussion of Scopes is found in Section 2.

The key information transmitted between Diameter peers is the current server load (to allow for better balancing of traffic, so as to preempt overload in the first place) as well as an indication of overload state and severity (overload information). The actual load and overload information is conveyed as a new compound AVP, added to any Diameter messages that allow for extensibility. As discussed in section 3.2 of [RFC6733], all CCFs are encouraged to include AVP-level extensibility by inclusion of a "*" [AVP] construct in their syntax definition. The document author has conducted an extensive (although admittedly not exhaustive) audit of existing applications, and found none lacking this property. The inclusion of load and overload information in existing messages has the property that the frequency with which information can be exchanged increases as load on the system goes up.

For the purpose of grouping the several different parts of load information together, this mechanism makes use of a Grouped AVP, called "Load-Info". The Load-Info AVP may appear one or more times in any extensible command, with the restriction that each instance of the Load-Info AVP must contain different Scopes.

Load and overload information can be conveyed during times of inter-node quiescence through the use of DWR/DWA exchanges. These exchanges can also be used to proactively change the overload or load level of a server when no other transaction is ready to be sent. Finally, in the unlikely event that an application is defined that precludes the inclusion of new AVPs in its commands, DWR/DWA exchanges can be sent at any rate acceptable to the server in order to convey load and overload information.

In [RFC3588], the DWR and DWA message syntax did not allow for the addition of new AVPs in the DWR and DWA messages. This oversight was fixed in [RFC6733]. To allow for transmission of load information on quiescent links, implementations of the mechanism described in this document are expected to correctly handle extension AVPs in DWR and DWA messages, even if such implementations have not otherwise been upgraded to support [RFC6733].

1.2. Overview of Operation

During the capabilities exchange phase of connection establishment, peers determine whether the connection will make use of the overload control mechanism; and, if so, which optional behaviors are to be employed.

The information sent between adjacent nodes includes two key metrics: Load (which, roughly speaking, provides a linear metric of how busy the node is), and Overload-Metric (which is input to the negotiated load abatement algorithm).

Message originators (whether originating a request or an answer) include one or more Load-Info AVPs in messages when they form them. These Load-Info AVPs reflect the originators' own load and overload state.

Because information is being used on a hop-by-hop basis, it is exchanged only between adjacent nodes. This means that any Diameter agent that forwards a message (request or answer) is required to remove any information received from the previous hop, and act upon it as necessary. Agents also add their own load and overload information (which may, at implementors' preference, take previous-hop information into account) into a new Load-Info AVP before sending the request or answer along.

Because the mechanism requires affirmative indication of support in the capabilities exchange phase of connection establishment, load and overload information will never be sent to intermediaries that do not support the overload mechanism. Therefore, no special provisions need to be made for removal of information at such intermediaries -- it will simply not be sent to them.

Message recipients are responsible for reading and acting upon load and overload information that they receive in such messages.

1.3. Documentation Conventions

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

2. Overload Scopes

In normal operation, a Diameter node may be overloaded for some but not all possible requests. For example, an agent that supports two realms (realm A and realm B in this example) may route traffic to one

set of servers for realm A, and another set of servers for realm B. If the realm A servers are overloaded but realm B servers are not, then the agent is effectively overloaded for realm A but not for realm B.

Despite the fact that Diameter agents can report on scopes that semantically map to constructs elsewhere in the network, it is important to keep in mind that overload state is still reported on a hop-by-hop basis. In other words, the overload state reported for realm A in the example above represents the aggregate of the agent's overload state along with the overload state being reported by applicable upstream servers (those serving realm A).

Even without the use of Diameter agents, similar situations may arise in servers that need to make use of external resources for certain applications but not for others. For example, if a single server is handling two applications, one of which uses an external database while the other does not, it may become overloaded for the application that uses the external database when the database response latency increases.

The indication of scopes for overload information (using the Overload-Info-Scope AVP; see Section 5.4) allows a node to indicate a subset of requests to which overload information is to be applied. This document defines seven scopes; only "Connection" scope is mandatory to implement. The use of the optional scopes, along with the use of any additional scopes defined in other documents, is negotiated at connection establishment time; see Section 3.1.

2.1. Scope Descriptions

Destination-Realm: This scope, which nodes **MUST** implement, pertains to all transactions that have a Destination-Realm AVP matching the indicated value.

Application-ID: This scope, which nodes **MUST** implement, pertains to all transactions that contain an Application-ID field matching the indicated value.

Destination-Host: This scope, which nodes **SHOULD** implement, pertains to all transactions that have a Destination-Host AVP matching the indicated value.

Host: This scope, which nodes **SHOULD** implement, pertains to all transactions sent directly to the host matching the indicated value.

Connection: This scope, which nodes MUST implement, pertains to all transactions sent on the same TCP connection or SCTP association. This scope has no details indicating which connection or association it applies to; instead, the recipient of an indication of "Connection" scope is to use the connection or association on which the message was received as the indicated connection or association. In other words, any use of Connection scope applies to "this connection."

Session-Group: This scope, which nodes MAY implement, pertains to all transactions in a session that has been assigned to the indicated group. For more information on assigning sessions to groups, see Section 3.6.

Session: This scope, which nodes MAY implement, pertains to all transactions in the indicated session.

Some applications do not have long-running sessions containing multiple transactions. For such applications, the use of "Session-Group" and "Session" scopes do not make sense. Such applications will instead make use of the most applicable of the remaining scopes (plus any negotiated extension scopes) to achieve overload control.

OPEN ISSUE: Is there value to including a stream-level scope for SCTP? We haven't been able to come up with a use case for doing so yet, but it wouldn't necessarily be unreasonable.

2.2. Combining Scopes

To allow for the expression of more complicated scopes than the primitives defined above, multiple Overload-Info-Scope AVPs may be included in a single Load-Info AVP. Semantically, these scopes are included in the following way:

- o Attributes of the different kinds are logically and-ed together (e.g., if both "Destination-Realm" and "Application-ID" are present, the information applies to requests sent that match both the realm and the application).
- o Attributes of the same kind are logically or-ed together (e.g., if two "Destination-Realm"s are present, the information applies to requests sent to either realm).
- o If a transaction falls within more than one scope, the "most overloaded" scope is used for traffic shaping.

To prevent the complexity of implementing arbitrary scope combination rules, only the following combinations of scopes are allowed (OPEN ISSUE -- we need to figure out what makes most sense for expressing these combinations. Formal grammar? Prose? A table of some kind? For now, they're expressed as a pseudo-ABNF):

- o 1*(Destination-Realm) 0*1(Application-ID)
- o 1*(Application-ID) 0*1(Destination-Realm)
- o 1*(Application-ID) 0*1(Destination-Host)
- o 1*(Application-ID) 0*1(Host)
- o 1*(Application-ID) 0*1(Connection)
- o 1*(Destination-Host)
- o 1*(Host)
- o 1*(Connection)
- o 1*(Session-Group) 0*1(Host | Connection)
- o 1*(Session) 0*1(Host | Connection)

OPEN ISSUE: Is this the right set of scope combinations? Is there a need for more? Are any of these unnecessary? Ideally, this should be the smallest set of combinations that lets nodes report what they realistically need to report.

Any document that creates additional scopes MUST define how they may be combined with all scopes registered with IANA at the time of their publication.

3. Diameter Node Behavior

The following sections outline the behavior expected of Diameter clients, servers, and agents that implement the overload control mechanism.

OPEN ISSUE: SIP Overload Control includes a sequence parameter to ensure that out-of-order messages do not cause the receiver to act on state that is no longer accurate. Is message reordering a concern in Diameter? That is, do we need to include sequence numbers in the messages to ensure that the receiver does not act on stale state information? Because Diameter uses only reliable, in-order transports, it seems that this isn't likely to be an issue. Is there room for a race when multiple connections are in use?

3.1. Connection Establishment Procedures

Negotiation for support of this mechanism is performed during Diameter capabilities exchange. Optional protocol features and extensions to this mechanism are also negotiated at this time. No

provision is provided for renegotiation of mechanism use or extensions during the course of a connection. If peers wish to make changes to the mechanism, they must create a new connection to do so.

The connection initiator includes a Load-Info AVP in the CER (Capabilities-Exchange-Request) message that it sends after establishing the connection. This Load-Info AVP MUST contain a Supported-Scopes AVP and an Overload-Algorithm AVP. The Supported-Scopes AVP includes a comprehensive list of scopes supported that the connection initiator can receive and understand. See Section 5.2 for information on the format of the Supported-Scopes AVP.

The Load-Info AVP in a CER message also MAY contain one or more Overload-Algorithm AVPs. If present, these AVPs indicate every Overload-Algorithm the connection initiator is willing to support for the connection that is being established. If the connection initiator supports only the "Loss" algorithm, it MAY indicate this fact by omitting the Overload-Algorithm altogether.

The Load-Info AVP in a CER message MAY also contain additional AVPs, as defined in other documents, for the purpose of negotiation extensions to the Overload mechanism.

The Diameter node that receives a CER message first examines it for the presence of a Load-Info AVP. If no such AVP is present, the node concludes that the overload control mechanism is not supported for this connection, and no further overload-related negotiation is performed. If the received CER contains a Load-Info AVP, the recipient of that message stores that information locally in the context of the connection being established. It then examines the Overload-Algorithm AVPs, if present, and selects a single algorithm from that list. If no Overload-Algorithm is indicated, then the base "Loss" algorithm is used for the connection. In either case, the recipient of the CER stores this algorithm in the context of the connection.

When a node conformant to this specification sends a Capabilities-Exchange-Answer (CEA) message in answer to a CER that contained a Load-Info AVP, the CEA MUST contain a Load-Info AVP. This Load-Info AVP MUST contain a Supported-Scopes AVP that includes a comprehensive list of scopes supported that the connection initiator can receive and understand. The CEA also contains zero or one Overload-Algorithm AVPs. If present, this Overload-Algorithm MUST match one of the Overload-Algorithm AVPs sent in the CER, and it indicates the overload control algorithm that will be used for the connection. If the CEA contains no Overload-Algorithm, the connection will use the "Loss" algorithm.

When a node receives a CEA message, it examines it for the presence of a Load-Info AVP. If no such AVP is present, the node concludes that the overload mechanism is not supported for this connection. If the received CEA contains a Load-Info AVP, then the recipient extracts the Supported-Scopes information, and stores them locally in the context of the connection being established. It then checks for the presence of an Overload-Algorithm AVP. If present, this AVP indicates the overload control algorithm that will be used for the connection. If absent, then the connection will use the "Loss" algorithm.

If a node receives a CEA message that indicates support for a scope that it did not indicate in its CER or which selects an overload control algorithm that it did not advertise in its CER, then it MUST terminate the connection by sending a DPR with a Disconnect-Cause of `NEGOTIATION_FAILURE`, (128 [actual value TBD]) indicating that the CEA sender has failed to properly follow the negotiation process described above.

Note that the Supported-Scopes announcement during capabilities exchange is a set of mutual advertisements of which scopes the two nodes are willing to receive information about. It is not a negotiation. It is perfectly acceptable for a node to send information for scopes it did not include in the Supported-Scopes AVP it sent, as long as the recipient indicated support for receiving such a scope. For example, a Diameter agent, during connection establishment with a client, may indicate support for receiving only "Connection" and "Host" scope; however, if the client indicated support for "Application" scope, then the agent is free to send Load-Info AVPs that make use of "Application" scope to the client.

3.2. Diameter Client and Diameter Server Behavior

The following sections describe the behavior that Diameter clients and Diameter servers implement for the overload control mechanism. Behavior at Diameter Agents is described in Section 3.3.

To implement overload control, Diameter nodes need to keep track of three important metrics for each of the scopes for which information has been received: the overload metric for the scope, the period of validity for that overload metric, and the load within that scope. Conceptually, these are data records indexed by the scope to which they apply. In the following sections, we refer to these data records with the term "scope entry." Further, when it is necessary to distinguish between those scope entries referring to the load information received from other nodes and those referring to the load information sent to other nodes, we use the term "remote scope entry" to refer to the information received from other nodes, and "local

scope entry" to refer to that information that is being maintained to send to other nodes.

In order to allow recipients of overload information to perform certain performance optimizations, we also define a new command flag, called 'O'verload. This bit, when set, indicates that the message contains at least one Load-Info AVP with a non-zero Overload-Metric -- in other words, the sending node is overloaded for at least one context. See Section 7.4 for the definition of the 'O'verload bit.

OPEN ISSUE: Is there anything we can do to make this 'O'verload bit even more useful? Perhaps setting it only when the overload value has changed, or changed by a certain amount?

3.2.1. Sending a Request to a Compliant Peer

This section applies only to those requests sent to peers who negotiated use of the overload control mechanism during capabilities exchange. Requests sent over other connections are handled the same as they would in the absence of the overload control mechanism.

Before sending a request, a Diameter node must first determine which scope applies. It does this as follows: first, a next hop host and connection are determined, according to normal Diameter procedures (potentially modified as described in Section 3.5.2). The sending node then searches through its list of remote scope entries (ignoring any whose Period-of-Validity has expired) to determine which ones match the combination of the fields in the current request, the next-hop host, and the selected connection. If none of the matching scope entries are in overload, then the message is handled normally, and no additional processing is required.

As an optimization, a sending node MAY choose to track whether any of its peers are in overload, and to skip the preceding step if it knows that no scopes are in overload.

If one or more matching scope entries are in overload, then the sending node determines which scope is most overloaded. The sending node then sends, drops, or otherwise modifies handling of the request according to the negotiated overload control algorithm, using the Overload-Metric from the selected scope entry as input to the algorithm.

When determining which requests are impacted by the overload control algorithm, request senders MAY take into account the type of message being sent and its contents. For example, messages within an existing session may be prioritized over those that create a new session. The exact rules for such prioritization will likely vary

from application to application. The authors expect that specifications that define or specify the use of specific Diameter Applications may choose to formally define a set of rules for such prioritization on a per-Application basis.

The foregoing notwithstanding, senders MUST NOT use the content or type of request to exempt that request from overload handling. For example, if a peer requests a 50% decrease in sent traffic using the "Loss" algorithm (see Section 4), but the traffic that the sending node wishes to send consists 65% of traffic that the sender considers critical, then the sender is nonetheless obliged to drop some portion of that critical traffic (e.g., it may elect to drop all non-critical traffic and 23% of the critical traffic, resulting in an overall 50% reduction).

The sending node then inserts one or more Load-Info AVPs (see Section 5.1) into the request. If the sender inserts more than one Load-Info AVP, then each Load-Info AVP MUST contain a unique scope, as specified by the Overload-Scope AVP(s) inside the Load-Info AVP.

Each Load-Info AVP in the request MUST contain an Overload-Metric (see Section 5.5), indicating whether (and to what degree) the sender is overloaded for the indicated scope. If this metric is not zero, then the Load-Info AVP MUST also contain a Period-Of-Validity AVP (see Section 5.6), indicating the maximum period the recipient should consider the Overload-Metric to be valid. Any message containing a non-zero Overload-Metric also MUST set the 'O'verload bit in the Command Flags field to indicate to the recipient that the message contains an overload indication. See Section 7.4 for the definition of the 'O'verload bit.

Each Load-Info AVP MUST also contain a Load AVP, indicating the server's load level within the context of the indicated scope. See Section 3.5.1 for details on generating this load metric. Note that a server's load may frequently be identical for all the scopes for which it sends information.

3.2.2. Receiving a Request

3.2.2.1. Receiving a Request from a Compliant Peer

A node that receives a request from a peer that has negotiated support for the overload control mechanism will extract the Load-Info AVPs from the request and use each of them to update its remote scope entries. First, the node attempts to locate an existing scope entry that corresponds to the Overload-Scope indicated in the Load-Info AVP. If one does not exist, it is created. The scope entry is then populated with the overload metric, period of validity, and load

information. The message is then processed as normal.

In some circumstances, request recipients can become sufficiently overloaded that even those messages received from complaint clients can overwhelm its processing capabilities. Under such circumstances, nodes MAY begin treating a subset of such requests as if they were received from noncompliant peers (as explained in the following section).

3.2.2.2. Receiving a Request from a Noncompliant Peer

An important aspect of the overload control mechanism is that Diameter nodes that do not implement the mechanism cannot have an advantage over those that do. In other words, it is necessary to prevent the situation that a network in overload will cease servicing those transactions from overload-compliant nodes in favor of those sent by those nodes that do not implement the overload control mechanism. To achieve this goal, message recipients need to track the overload control metric on behalf of those sending nodes that do not implement overload, and to reject messages from those nodes that would have been dropped if the sender had implemented the overload mechanism.

A node that receives a request from a peer that has not negotiated support for the overload control mechanism searches through its list of local scope entries to determine which ones match the combination of the fields in the received request. (These are the entries that indicate the Overload-Metric that the node would have sent to the peer if the peer had supported the overload mechanism). If none of the matching scope entries are in overload, then the message is sent normally, and no additional processing is required.

If one or more matching local scope entries are in overload, then the node determines which scope is most overloaded. The node then executes the "Loss" overload control algorithm (see Section 4) using the overload metric in that most overloaded scope. If the result of running that algorithm determines that a sender who had implemented the overload control mechanism would have dropped the message, then the recipient MUST reply to the request with a `DIAMETER_PEER_IN_OVERLOAD` response (see Section 7.3).

3.2.3. Sending an Answer to a Compliant Peer

This section applies only to those answers sent to peers who negotiated use of the overload control mechanism during capabilities exchange.

When sending an answer, a Diameter node inserts one or more Load-Info

AVPs (see Section 5.1) into the answer. If the sender inserts more than one Load-Info AVP, then each Load-Info AVP MUST contain a unique scope, as specified by the Overload-Scope AVP(s) inside the Load-Info AVP.

Each Load-Info AVP in the answer MUST contain an Overload-Metric (see Section 5.5), indicating whether (and to what degree) the server is overloaded for the indicated scope. If this metric is not zero, then the Load-Info AVP MUST also contain a Period-Of-Validity AVP (see Section 5.6), indicating the maximum period the recipient should consider the Overload-Metric to be valid. Any message containing a non-zero Overload-Metric also MUST set the 'O'verload bit in the Command Flags field to indicate to the recipient that the message contains an overload indication. See Section 7.4 for the definition of the 'O'verload bit.

It is important to note that using this mechanism creates a closed feedback loop, with some amount of lag introduced by overload processing and the network. As such, implementers must be aware of the potential for such a system to produce oscillations in the overload level. Without proper control, it is also possible for these oscillations to diverge, resulting in undesirable behavior. There are several ways to address this issue, and it is left to implementors to determine the best way for their particular situation. However, at a minimum senders of overload control information SHOULD apply hysteresis to the Overload-Metric, and signal easing of overload more slowly than signaling increases.

Each Load-Info AVP MUST also contain a Load AVP, indicating the server's load level within the context of the indicated scope. See Section 3.5.1 for details on generating this load metric. Note that a server's load may frequently be identical for all the scopes for which it sends information.

3.2.4. Receiving an Answer from a Compliant Peer

A node that receives an answer from a peer that has negotiated support for the overload control mechanism will extract the Load-Info AVPs from the answer and use each of them to update its remote scope entries. First, the node attempts to locate an existing scope entry that corresponds to the Overload-Scope indicated in the Load-Info AVP. If one does not exist, it is created. The scope entry is then populated with the overload metric, period of validity, and load information. The message is then processed as normal.

3.3. Diameter Agent Behavior

This section discusses the behavior of a Diameter Agent acting as a Proxy or Relay. Diameter Agents that provide redirect or translation services behave the same as Diameter Servers for the purpose of overload control, and follow the procedures defined in Section 3.2.

Whenever sending a request or an answer, Agents MUST include a Load-Info AVP reflecting the Agent's overload and load information. In formulating this information, the Agent may choose to use only that information relating to its own local resources. However, better network behavior can be achieved if agents incorporate information received from their peers when generating overload information. The exact means for incorporating such information is left to local policy at the agent.

For example: consider an agent that distributes sessions and transactions among three Diameter servers, each hosting a different Diameter application. While it would be compliant for the Agent to only report its own overload state (i.e., at "Host" scope), overall network behavior would be improved if it chose to also report overload state for up to three additional scopes (i.e. at "Application-ID" scope), incorporating the Overload information received from each server in these scopes.

3.3.1. Proxying a Request

Upon receiving a request, a Diameter Proxy or Relay performs the steps detailed in Section 3.2.2.

The agent then MUST remove all Load-Info AVPs from the request: Load-Info is never passed through a Proxy or Relay transparently.

When the Diameter Agent proxies or relays a request, it follows the process outlined in Section 3.2.1.

3.3.2. Proxying an Answer

Upon receiving an answer, a Diameter Agent follows the process described in Section 3.2.4 to update its remote scope entries.

The Agent then MUST remove all Load-Info AVPs from the answer: Load-Info is never passed through a Proxy or Relay transparently.

When the Diameter Agent proxies or relays a response, it follows the process outlined in Section 3.2.3.

3.4. Proactive Load and Overload Communication

Because not all Diameter links will have constant traffic, it may be occasionally necessary to send overload and/or load information over links that would otherwise be quiescent. To proactively send such information to peers, the Diameter node with information to convey may choose to send a Diameter Watchdog Request (DWR) message to its peers. The procedure described in Section 3.2.1 applies to these requests, which provides the means to send load and overload information.

In order to prevent unnecessarily diminished throughput between peers, a Diameter node SHOULD proactively send a DWR to all its peers whenever it leaves an overload state. Similarly, in order to provide peers the proper data for load distribution, nodes SHOULD send DWR messages to a peer if the load information most recently sent to that peer has changed by more than 20% and is more than 5 seconds old.

3.5. Load Processing

While the remainder of the mechanism described in this document is aimed at handling overload situations once they occur, it is far better for a system if overload can be avoided altogether. In order to facilitate overload avoidance, the overload mechanism includes the ability to convey node load information.

Semantically, the Load information sent by a Diameter node indicates the current utilization of its most constrained resource. It is a linear scale from 0 (least loaded) to 65535 (most loaded).

It is critical to distinguish between the value conveyed in the Load AVP and the value conveyed in the Overload-Metric AVP. The Load AVP is computed and used independent of the Overload-Algorithm selected for a connection, while the Overload-Metric is meaningful only in the context of the selected algorithm. Most importantly, the Load information never has any impact on the behavior specified in the overload algorithm. If a node reports a Load of 65535, but the Overload-Metric does not indicate any need to apply the selected overload control algorithm, then the sender MUST NOT apply the selected overload control algorithm. Conversely, if a node is reporting an Overload-Metric that requires the recipient to take action to reduce traffic, those actions MUST be taken, even if the node is simultaneously reporting a Load value of 0.

3.5.1. Sending Load Information

Diameter nodes implementing the overload mechanism described in this document MUST include a Load AVP (inside a Load-Info AVP) in every

Diameter message (request and answer) they send over a connection that has been negotiated to use the overload control mechanism. Note that this requirement does not necessitate calculation of the Load metric each time a message is sent; the Load value may be calculated periodically (e.g., every 100 ms), and used for every message sent until it is recalculated.

The algorithm for generation of the load metric is a matter of local policy at the Diameter node, and may vary widely based on the internal software architecture of that node.

For advanced calculations of Load, anticipated inputs to the computation include CPU utilization, network utilization, processor interrupts, I/O throughput, and internal message queue depths.

To free implementors from the potential complexity of determining an optimal calculation for load, we define a very simple, baseline load calculation that MAY be used for the purpose of populating the Load AVP. Implementations using this simplified calculation will use a configured, hard-coded, or Service Level Agreement (SLA)-defined maximum number of transactions per second (TPS) which a node is known to be able to support without issue. These implementations simply report their load as a linear representation of how much of this known capacity is currently in use:

$$\text{Load} = \text{MIN}(\text{Current_TPS} * 65535 / \text{Maximum_TPS}, 65535)$$

To prevent rapid fluctuations in the load metric, nodes SHOULD report a rolling average of the calculated load rather than the actual instantaneous load at any given moment.

Load information is scoped to the level indicated by the Overload-Info-Scope AVP present in the Load-Info AVP in which the Load AVP appears.

3.5.2. Receiving Load Information

While sending load information is mandatory, the actual processing of load information at a recipient is completely optional. Ideally, recipients will use the load information as input to a decision regarding which of multiple equivalent servers to use when initiating a new connection. Recipients may choose to update load information on receipt of every message; alternately, they may periodically "sample" messages from a host to determine the load it is currently reporting.

3.5.2.1. Example Load Handling

This section describes a non-normative example of how recipients can use Load information received from other Diameter nodes. At a high level, the concept is that received load metrics are used to scale the distribution algorithm that the node uses for selection of a server from a group of equivalent servers.

Consider a client that uses DNS to resolve a host name into IP addresses. In this example, the client is attempting to reach the server for the realm example.com. It performs a NAPTR query for the "AAA+D2T" record for that domain, and receives a result pointing to the SRV record "_diameter._tcp.example.com". Querying for this SRV record, in turn, results in three entries, with the same priorities:

SRV Weight	Server Name
20	server-a.example.com
20	server-b.example.com
60	server-c.example.com

The client then examines the currently reported loads for each of the three servers. In this example, we are asserting that the reported load metrics are as follows:

Load	Server Name
13107 (20%)	server-a.example.com
26214 (60%)	server-b.example.com
52428 (80%)	server-c.example.com

Based on this load information, the client scales the SRV weights proportional to each server's reported load; the general formula is:

$$\text{new_weight} = \text{original_weight} * (65535 - \text{load}) / 65535$$

The node then calculates a new set of weights for the destination hosts:

- o server-a: new_weight = 20 * (65535 - 13107) / 65535 = 16
- o server-b: new_weight = 20 * (65535 - 26214) / 65535 = 12
- o server-c: new_weight = 60 * (65535 - 52428) / 65535 = 12

These three new weights (16, 12, and 12) are then used as input to

the random selection process traditionally used when selecting among several SRV records.

Note that this example is provided in the context of DNS SRV processing; however, it works equally well in the case that server processing weights are provisioned or made available through an alternate resolution process.

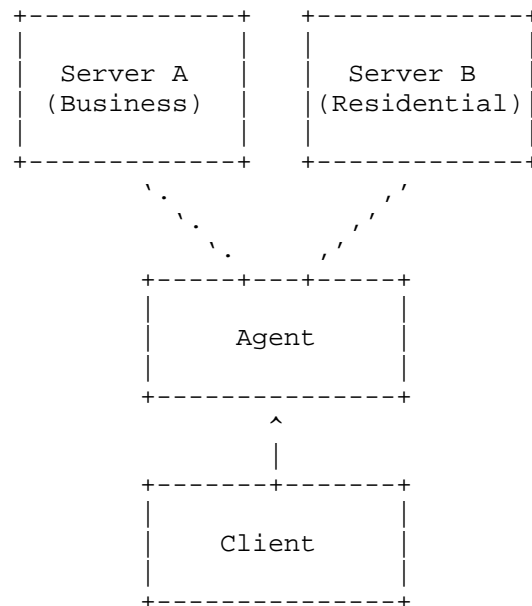
3.6. Session Establishment for Session Groups

The procedure in this section applies to any Diameter operation that may result in the creation of a new Diameter session. Note that these operations are performed in addition to any normal message processing, and in addition to the operations described in the following sections.

3.6.1. Session Group Concepts

At the time a session is established, the server and/or the client may choose to assign the newly created session to a Session Group that they can use to refer to the session (and other sessions in the same group) in later overload-related messages. This grouping is intended to be used by servers that have visibility into resources that may be independently overloaded, but which do not correspond to an existing Diameter construct (such as Application, Realm, or Destination Server).

One example of a server having visibility into resources that don't have a corresponding Diameter construct is a Diameter Agent servicing a mixed community of users -- say, one authenticated by a "Business" server, and another authenticated by a "Residential" server. The client in this network does not know which group any given session belongs in; the routing of sessions is based on information available only to the agent.



In this case, the Agent may wish to assign sessions to two client-visible Session Groups when the session is established. By doing so, the Agent gains the ability to report Load and Overload metrics to the Client independently for the two classes of users. This can be extremely helpful, for example, in allowing the Agent to ask the Client to throttle traffic for the Residential server when it becomes overloaded, without impacting sessions pertaining to the Business server.

Similar situations can arise even without the presence of Diameter Agents in the network: a server may have a class of sessions that require access to an off-board database (which can, itself, become overloaded), while also servicing a class of sessions that is handled entirely by a local authentication table. The server can use Session Groups to assign these two classes of sessions to different groups, and report overload on the class using the (overloaded) off-board database without impacting the other sessions.

In some applications, it is possible to have the session established by one peer (e.g., in the upstream direction), while some subsequent in-session transactions are initiated by the other peer (e.g., in the downstream direction). Because of this possibility, the overload mechanism allows both peers to establish a Session Group at the time the session is set up. The session identifiers are scoped to the node that sends them. In other words, if a server assigns a session to a group called "Residential", this group is not related to a

client group (if any) by the same name. For clarity, this document will refer to the session group assigned by the server performing the processing as a "local session group," and the session group assigned by the remote node as a "remote session group."

Nodes that send a session-creating request follow normal Diameter procedures, along with the additional behavior described in Section 3.2.1 and Section 3.3.1, as appropriate. Such nodes may also assign the session to a Session Group, as long as the peer to which they are communicating indicated support for the "Session-Group" scope during capabilities exchange. Whether to do so and what group to assign a session to is done according to local policy. To perform such assignment, the node will include a Session-Group AVP (see Section 5.7 in the Load-Info AVP for the session creating request. These nodes also store the assigned name as the session's local session group.

3.6.2. Session Group Procedures

The procedures in this section only apply on connections for which support for the "Session-Group" scope has been negotiated during capabilities exchange. See Section 3.1.

When a node receives a session creating request, it MUST check that request for the presence for a Session-Group AVP in its Load-Info AVP. If one is present, it stores that session group name as the remote session group name for that server. This allows clients to assign the session to a group, allowing it to indicate overload for server-initiated transactions in the resulting session.

When a node replies to a session creating request, it can choose to assign the newly-established session to a session group. Whether it chooses to do so is independent of whether the remote node assigned the session to a session group. To perform such an assignment, the node includes a Session-Group AVP in the Load-Info AVP sent in answer to the session-creating request. These nodes also store the assigned name as the session's local session group.

Finally, when a node that has sent a session-creating request receives a corresponding answer message, it MUST check that answer for the presence of a Session-Group AVP in its Load-Info AVP. If one is present, it stores that session group name as the remote session group name for that server.

4. Loss-Based Overload Control Algorithm

This section describes a baseline, mandatory-to-implement overload

control algorithm, identified by the indicator "Loss". This algorithm allows a Diameter peer to ask its peers to reduce the number of requests they would ordinarily send by a specified percentage. For example, if a peer requests of another peer that it reduce the traffic it is sending by 10%, then that peer will redirect, reject, or treat as failed, 10% of the traffic that would have otherwise been sent to this Diameter node.

4.1. Overload-Metric values for the 'Loss' Algorithm

A Diameter node entering the overload state for any of the scopes that it uses with its peers will calculate a value for its Overload Metric, in the range of 0 to 100 (inclusive). This value indicates the percentage traffic reduction the Diameter node wishes its peers to implement. The computation of the exact value for this parameter is left as an implementation choice at the sending node. It is acceptable for implementations to request different levels of traffic reduction to different peers according to local policy at the Diameter node. These Overload Metrics are then communicated to peers using the Overload-Metric AVP in requests and answers sent by this node.

Recipients of Overload-Metric AVPs on connections for which the "Loss" algorithm has been specified MUST reduce the number of requests sent in the corresponding scope by that percentage, either by redirecting them to an alternate destination, or by failing the request. For a Diameter Agent, these failures are indicated to the peer who originated the request by sending a `DIAMETER_PEER_IN_OVERLOAD` response (see Section 7.3). For diameter clients, these failures cause the client to behave as if they received a transient error in response to the request.

It is acceptable, when implementing the "Loss" algorithm, for the reduction in transactions to make use of a statistical loss function (e.g., random assignment of transactions into "success" and "failure" categories based on the indicated percentage). In such a case, the actual traffic reduction might vary slightly from the percentage indicated, albeit in an insignificant amount.

The selection of which messages to withhold from sending does not need to be arbitrary. For example, implementations are allowed to distinguish between higher-priority and lower-priority messages, and drop the lower-priority messages in favor of dropping the higher priority messages, as long as the total reduction in traffic conforms to the Overload-Metric in effect at the time. The selection of which messages to prioritize over others will likely vary from application to application (and may even be subject to standardization as part of the application definition). One example of such a prioritization

scheme would be to treat those messages that result in the creation of a new session as lower priority than those messages sent in the context of an established session.

4.2. Example Implementation

The exact means a client uses to implement the requirement that it reduce traffic by a requested percentage is left to the discretion of the implementor. However, to aid in understanding the nature of such an implementation, we present an example of a valid implementation in pseudo-code.

In this example, we consider that the sending node maintains two classes of request. The first category are considered of lower priority than the second category. If a reduction in traffic is required, then these lower priority requests will be dropped before any of the higher priority requests are dropped.

The sending Diameter node determines the mix of requests falling into the first category, and those falling into the second category. For example, 40% of the requests may be in the lower-priority category, while 60% are in the higher-priority category.

When a node receives an overload indication from one of its peers, it converts the Overload-Metric value to a value that applies to the first category of requests. For example, if the Overload-Metric for the applicable context is "10", and 40% of the requests are in the lower-priority category, then:

$$10 / 40 * 100 = 25$$

Or 25% of the requests in the first category can be dropped, with an overall reduction in sent traffic of 10%. The sender then drops 25% of all category 1 requests. This can be done stochastically, by selecting a random number for each sent packet between 1 to 100 (inclusive), and dropping any packet for which the resulting percentage is equal to or less than 25. In this set of circumstances, messages in the second category do not require any reduction to meet the requirement of 25% traffic reduction.

A reference algorithm is shown below, using pseudo-code.

```
cat1 := 80.0           // Category 1 --- subject to reduction
cat2 := 100.0 - cat1 // Category 2 --- Under normal operations
// only subject to reduction after category 1 is exhausted.
// Note that the above ratio is simply a reasonable default.
// The actual values will change through periodic sampling
// as the traffic mix changes over time.
```

```
while (true) {
    // We're modeling message processing as a single work queue
    // that contains both incoming and outgoing messages.
    msg := get_next_message_from_work_queue()

    update_mix(cat1, cat2) // See Note below

    switch (msg.type) {

    case outbound request:
        destination := get_next_hop(msg)
        oc_context := get_oc_scope(destination,msg)

        if (we are in overload) {
            add_overload_avps(msg)
        }

        if (oc_context == null) {
            send_to_network(msg) // Process it normally by sending the
            // request to the next hop since this particular
            // destination is not subject to overload
        }
        else {
            // Determine if server wants to enter in overload or is in
            // overload
            in_oc := extract_in_oc(oc_context)

            oc_value := extract_oc(oc_context)
            oc_validity := extract_oc_validity(oc_context)

            if (in_oc == false or oc_validity is not in effect) {
                send_to_network(msg) // Process it normally by sending
                // the request to the next hop since this particular
                // destination is not subject to overload. Optionally,
                // clear the oc context for this server (not shown).
            }
            else { // Begin perform overload control
                r := random()
                drop_msg := false

                if (cat1 >= cat2) {
                    category := assign_msg_to_category(msg)
                    pct_to_reduce_cat2 := 0
                    pct_to_reduce_cat1 := oc_value / cat1 * 100
                    if (pct_to_reduce_cat1 > 100) {
                        // Get remaining messages from category 2
                        pct_to_reduce_cat2 := 100 - pct_to_reduce_cat1
                        pct_to_reduce_cat1 := 100
                    }
                }
            }
        }
    }
```

```
    }

    if (category == cat1) {
        if (r <= pct_to_reduce_cat1) {
            drop_msg := true
        }
    }
    else { // Message from category 2
        if (r <= pct_to_reduce_cat2) {
            drop_msg := true
        }
    }
}
else { // More category 2 messages than category 1;
      // indicative of an emergency situation. Since
      // there are more category 2 messages, don't
      // bother distinguishing between category 1 or
      // 2 --- treat them equal (for simplicity).
    if (r <= oc_value)
        drop_msg := true
}

if (drop_msg == false) {
    send_to_network(msg) // Process it normally by
    // sending the request to the next hop
}
else {
    // Do not send request downstream, handle locally by
    // generating response (if a proxy) or treating as
    // an error (if a user agent).
}
} // End perform overload control
}

end case // outbound request

case outbound answer:
    if (we are in overload) {
        add_overload_avps(msg)
    }
    send_to_network(msg)

end case // outbound answer

case inbound answer:
    create_or_update_oc_scope() // For the specific server
    // that sent the answer, create or update the oc scope;
    // i.e., extract the values of the overload AVPs
```

```

        // and store them in the proper scopes for later use.
        process_msg(msg)

    end case // inbound answer
    case inbound request:
        create_or_update_oc_scope()

        if (we are not in overload) {
            process_msg(msg)
        }
        else { // We are in overload
            if ( connection supports overload)
                process_msg(msg)
            }
            else { // Sender does not support oc
                if (local_policy(msg) says process message) {
                    process_msg(msg)
                }
                else {
                    send_answer(msg, DIAMETER_PEER_IN_OVERLOAD)
                }
            }
        }
    end case // inbound request
}

```

A simple way to sample the traffic mix for category 1 and category 2 is to associate a counter with each category of message. Periodically (every 5-10s), get the value of the counters and calculate the ratio of category 1 messages to category 2 messages since the last calculation.

Example: In the last 5 seconds, a total of 500 requests were scheduled to be sent. Assume that 450 out of 500 were messages subject to reduction and 50 out of 500 were classified as requests not subject to reduction. Based on this ratio, cat1 := 90 and cat2 := 10, or a 90/10 mix will be used in overload calculations.

Of course, this scheme can be generalized to include an arbitrary number of priorities, depending on how many different classes of messages make sense for the given application.

5. Diameter AVPs for Overload

NOTE: THE AVP NUMBERS IN THIS SECTION ARE USED FOR EXAMPLE PURPOSES ONLY. THE FINAL AVP CODES TO BE USED WILL BE ASSIGNED BY IANA DURING THE PUBLICATION PROCESS, WHEN AND IF THIS DOCUMENT IS PUBLISHED AS AN RFC.

Attribute Name	AVP Code	Sec. Def.	Data Type	MUST	MUST NOT
Load-Info	1600	5.1	Grouped		M,V
Supported-Scopes	1601	5.2	Unsigned64		M,V
Overload-Algorithm	1602	5.3	Enumerated		M,V
Overload-Info-Scope	1603	5.4	OctetString		M,V
Overload-Metric	1604	5.5	Unsigned32		M,V
Period-Of-Validity	1605	5.6	Unsigned32		M,V
Session-Group	1606	5.7	UTF8String		M,V
Load	1607	5.8	Unsigned32		M,V

5.1. Load-Info AVP

The Load-Info AVP (AVP code 1600) is of type Grouped, and is used as a top-level container to group together all information pertaining to load and overload information. Every Load-Info AVP MUST contain one Overload-Information-Scope AVP, and one Overload-Metric AVP.

The Grouped Data field of the Load-Info AVP has the following CCF grammar:

```

< Load-Info > ::= < AVP Header: 1600 >
                  < Overload-Metric >
                  * { Overload-Info-Scope }
                  [ Supported-Scopes ]
                  * [ Overload-Algorithm ]
                  [ Period-Of-Validity ]
                  [ Session-Group ]
                  [ Load ]
                  * [ AVP ]

```

5.2. Supported-Scopes AVP

The Supported-Scopes AVP (AVP code 1601) is of type Uint64, and is used during capabilities exchange to indicate the scopes that a given node can receive on the connection. Nodes that support the mechanism defined in this document MUST include a Supported-Scopes AVP in all CER messages. It also MUST appear in any CEA messages sent in answer

to a CER message containing a Load-Info AVP. The Supported-Scopes AVP MUST NOT appear in any other message types. See Section 5.4 for an initial list of scopes.

The Supported-Scopes AVP contains a bitmap that indicates the scopes supported by the sender. Within the bitmap, the least significant bit indicates support for scope 1 (Destination-Realm), while the next least significant bit indicates support for scope 2 (Application-ID), and so on. In general, if we consider the bits to be numbered from 0 (LSB) to 63 (MSB), then any bit n corresponds to the scope type numbered $n+1$. This scheme allows for up to 64 total scopes to be supported. More formally, the bitmask used to indicate support for any specific context is calculated as follows (where the symbol " \ll " indicates a bit shift left):

$$\text{bitmask} = 1 \ll (n - 1)$$

For additional clarity, the bitmasks for the scopes defined in this document are as follows:

Scope	Bitmask	Scope
1	0x0000000000000001	Destination-Realm
2	0x0000000000000002	Application-ID
3	0x0000000000000004	Destination-Host
4	0x0000000000000008	Host
5	0x0000000000000010	Connection
6	0x0000000000000020	Session-Group
7	0x0000000000000040	Session

The advertisement process that makes use of the Supported-Scopes AVP is described in Section 3.1.

5.3. Overload-Algorithm AVP

The Overload-Algorithm AVP (AVP code 1602) is of type Enumerated, and is used to negotiate the algorithm that will be used for load abatement. The Overload-Algorithm AVP MAY appear in CER and CEA messages, and MUST NOT appear in any other message types. If absent, an Overload Algorithm of type 1 (Loss) is indicated. Additional values can be registered by other documents; see Appendix C.1. Initial values for the enumeration are as follows:

AVP Values	Attribute Name	Reference
0	Reserved	-
1	Loss	[RFC xxxx]

5.4. Overload-Info-Scope AVP

The Overload-Info-Scope AVP (AVP code 1603) is of type OctetString, and is used to indicate to which scope the Overload-Metric applies.

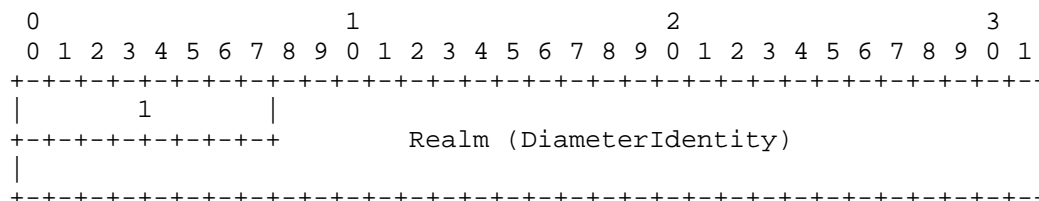
See Section 2 for a definition of the different scope types and a formal description of how they are applied. Other documents may define additional scopes; see Appendix C.2 for details.

0										1										2										3									
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1								
Scope										Details																													

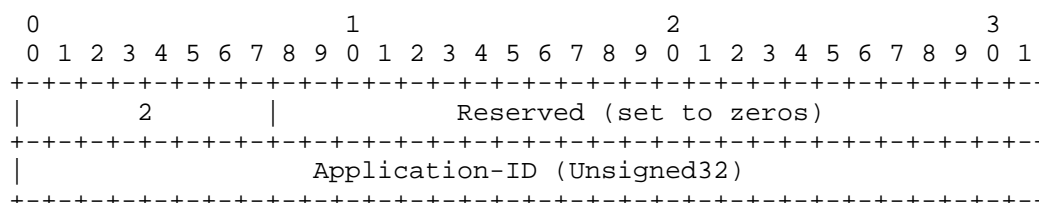
Scope	Attribute Name	Reference
0	Reserved	[RFC xxxx]
1	Destination-Realm	[RFC xxxx]
2	Application-ID	[RFC xxxx]
3	Destination-Host	[RFC xxxx]
4	Host	[RFC xxxx]
5	Connection	[RFC xxxx]
6	Session-Group	[RFC xxxx]
7	Session	[RFC xxxx]

Each Overload-Info-Scope has a different encoding, according to the identifier used to designate the corresponding scope. The formats for the seven scopes defined in this document are given in the following section.

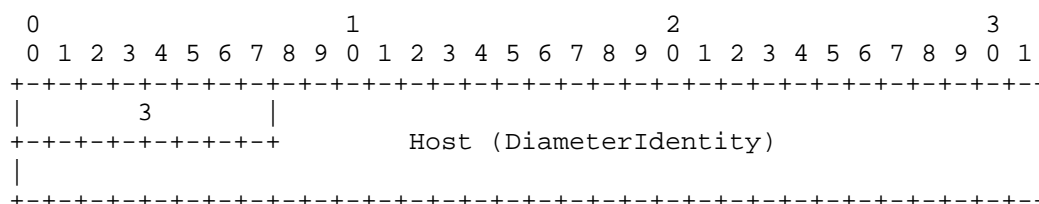
5.4.1. Realm Scope



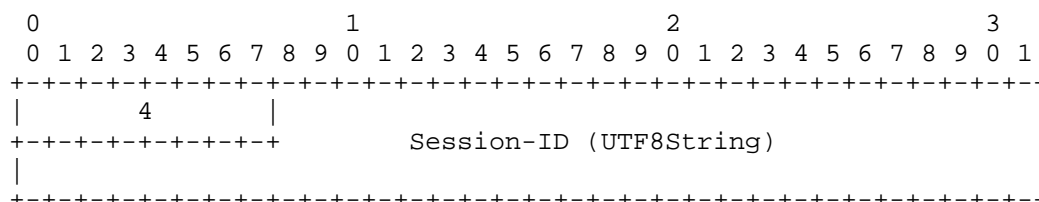
5.4.2. Application-ID Scope



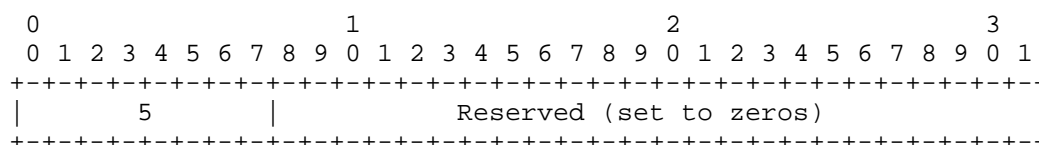
5.4.3. Host Scope



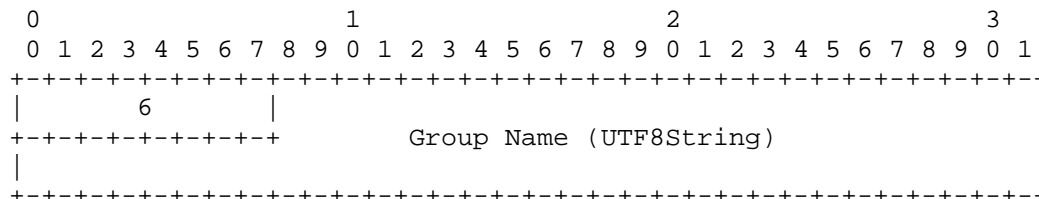
5.4.4. Session Scope



5.4.5. Connection Scope



5.4.6. Session Group Scope



5.5. Overload-Metric AVP

The Overload-Metric AVP (AVP code 1604) is of type Unsigned32, and is used as input to the load mitigation algorithm. Its definition and interpretation is left up to each individual algorithm, with the exception that an Overload-Metric of "0" always indicates that the node is not in overload (that is, no load abatement procedures are in effect) for the indicated scope.

5.6. Period-Of-Validity AVP

The Period-Of-Validity AVP (AVP code 1605) is of type Unsigned32, and is used to indicate the length of time, in seconds, the Overload-Metric is to be considered valid (unless overridden by a subsequent Overload-Metric in the same scope). It MUST NOT be present if the Overload-Metric is '0', and MUST be present otherwise.

5.7. Session-Group AVP

The Session-Group AVP (AVP code 1606) is of type UTF8String, and is used to assign a new session to the session group that it names. The Session-Group AVP MAY appear once in the answer to a session-creating request, and MUST NOT appear in any other message types.

5.8. Load AVP

The Load AVP (AVP code 1607) is of type Unsigned32, and is used to indicate the load level of the scope in which it appears. See Section 3.5 for additional information.

6. Security Considerations

A key concern for recipients of overload metrics and load information is whether the peer from which the information has been received is authorized to speak for the indicated scope. For scopes such as "Host" and "Connection", such authorization is obvious. For other scopes, such as "Application-ID" and "Realm", the potential for a

peer to maliciously or accidentally reduce traffic to a third party is evident. Implementations may choose to ignore indications from hosts which do not clearly have authority over the indicated scope; alternately, they may wish to further restrict the scope to apply only to the host from which the information has been received.

On the other hand, multiple nodes that are under the same administrative control (or a tightly controlled confederation of control) may be implicitly trusted to speak for all scopes within that domain of control. Implementations are encouraged to allow configuration of inherently trusted servers to which the foregoing restrictions are not applied.

Open Issue: There are almost certainly other security issues to take into consideration here. For example, we might need to include guidance around who gets to see our own load information, and potentially changing the granularity of information presented based on trust relationships.

7. IANA Considerations

This document defines new entries in several existing IANA tables. It also creates two new tables.

7.1. New Diameter AVPs

The following entries are added to the "AVP Codes" table under the "aaa-parameters" registry.

AVP Code	Attribute Name	Reference
1600	Load-Info	RFC xxxx
1601	Supported-Scopes	RFC xxxx
1602	Overload-Algorithm	RFC xxxx
1603	Overload-Info-Scope	RFC xxxx
1604	Overload-Metric	RFC xxxx
1605	Period-Of-Validity	RFC xxxx
1606	Session-Group	RFC xxxx
1607	Load	RFC xxxx

7.2. New Diameter Disconnect-Cause

The following entry is added to the "Disconnect-Cause AVP Values (code 273)" table in the "aaa-parameters" registry:

AVP Values	Attribute Name	Reference
128 [actual value TBD]	NEGOTIATION_FAILURE	RFC xxxxx

7.3. New Diameter Response Code

The following entry is added to the "Result-Code AVP Values (code 268) - Transient Failures" table in the "aaa-parameters" registry:

AVP Values	Attribute Name	Reference
4128 [actual value TBD]	DIAMETER_PEER_IN_OVERLOAD	RFC xxxxx

7.4. New Command Flag

The following entry is added to the "Command Flags" table in the "aaa-parameters" registry:

bit	Name	Reference
4	'O'verload	RFC xxxxx

7.5. Overload Algorithm Registry

This document defines a new table, to be titled "Overload-Algorithm Values (code 1602)", in the "aaa-parameters" registry. Its initial values are to be taken from the table in Section 5.3.

New entries in this table follow the IANA policy of "Specification Required." (Open Issue: The WG should discuss registration policy to ensure that we think this is the right balance).

7.6. Overload Scope Registry

This document defines a new table, to be titled "Overload-Info-Scope Values (code 1603)", in the "aaa-parameters" registry. Its initial values are to be taken from the table in Section 5.4.

New entries in this table follow the IANA policy of "Specification Required." (Open Issue: The WG should discuss registration policy to ensure that we think this is the right balance).

8. References

8.1. Normative References

- [I-D.ietf-dime-overload-reqs]
McMurry, E. and B. Campbell, "Diameter Overload Control Requirements", draft-ietf-dime-overload-reqs-06 (work in progress), April 2013.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.

8.2. Informative References

- [I-D.ietf-soc-overload-control]
Gurbani, V., Hilt, V., and H. Schulzrinne, "Session Initiation Protocol (SIP) Overload Control", draft-ietf-soc-overload-control-12 (work in progress), February 2013.
- [I-D.ietf-soc-overload-rate-control]
Noel, E. and P. Williams, "Session Initiation Protocol (SIP) Rate Control", draft-ietf-soc-overload-rate-control-04 (work in progress), April 2013.
- [RFC3588] Calhoun, P., Loughney, J., Guttman, E., Zorn, G., and J. Arkko, "Diameter Base Protocol", RFC 3588, September 2003.
- [RFC6357] Hilt, V., Noel, E., Shen, C., and A. Abdelal, "Design Considerations for Session Initiation Protocol (SIP) Overload Control", RFC 6357, August 2011.

Appendix A. Acknowledgements

This work was inspired by and borrows heavily from the SIP overload control mechanism described in [I-D.ietf-soc-overload-control]. The authors of this document are deeply grateful to the editor and authors of that work, as well as its many contributors.

Thanks to Ben Campbell for significant input to the initial mechanism design. The author also thanks Martin Dolly, Bob Wallace, John Gilmore, Matt McCann, Jonathan Palmer, Kedar Karmarkar, Imtiaz Shaikh, Jouni Korhonen, Uri Baniel, Jianrong Wang, Brian Freeman, and

Eric Noel for early feedback on the mechanism.

Appendix B. Requirements Analysis

This section analyzes the mechanism described in this document against the set of requirements detailed in [I-D.ietf-dime-overload-reqs].

REQ 1: The overload control mechanism MUST provide a communication method for Diameter nodes to exchange load and overload information.

Compliant. The mechanism uses new AVPs piggybacked on existing Diameter messages to exchange load and overload information.

REQ 2: The mechanism MUST allow Diameter nodes to support overload control regardless of which Diameter applications they support. Diameter clients must be able to use the received load and overload information to support graceful behavior during an overload condition. Graceful behavior under overload conditions is best described by REQ 3.

Compliant. Piggybacked AVPs conveying overload control information is sent on every Diameter message to compliant peers, without regard to its Application-ID. The use of the Application-ID scope allows information relevant to one application to be piggybacked on messages for other applications.

Information sent to peers includes load and overload information for use by overload control algorithms, intended for graceful overload mitigation. The mechanism is hop-by-hop and has provisions for agents to forward or aggregate load and overload information towards clients and servers so that each element can have appropriate information for graceful overload control.

REQ 3: The overload control mechanism MUST limit the impact of overload on the overall useful throughput of a Diameter server, even when the incoming load on the network is far in excess of its capacity. The overall useful throughput under load is the ultimate measure of the value of an overload control mechanism.

Compliant. The mechanism provides information nodes use to affect the impacts of overload according to agreed upon algorithms. By controlling or reducing traffic sent towards overloaded elements, using overload control information as described in the mechanism, the effects of overload can be limited. Use of scopes provides a means to minimize the impact of overload mitigation, increasing overall useful throughput during overload conditions.

- REQ 4: Diameter allows requests to be sent from either side of a connection and either side of a connection may have need to provide its overload status. The mechanism **MUST** allow each side of a connection to independently inform the other of its overload status.

Compliant. Overload control information can be piggybacked on any Diameter message. This applies for requests and answers sent from either side of a connection.

- REQ 5: Diameter allows nodes to determine their peers via dynamic discovery or manual configuration. The mechanism **MUST** work consistently without regard to how peers are determined.

Compliant. The mechanism makes no assumptions as to how peers are determined. Discovery of supporting peers is accomplished as part of the normal capabilities exchange and does not affect how or where these exchanges occur.

- REQ 6: The mechanism designers **SHOULD** seek to minimize the amount of new configuration required in order to work. For example, it is better to allow peers to advertise or negotiate support for the mechanism, rather than to require this knowledge to be configured at each node.

Compliant. The mechanism adds information to the existing Diameter capabilities exchange mechanism for determining peer support and overload control characteristics. Since this is accomplished dynamically at the start of connections, no provisioning is required to establish which peers support the mechanism and in what fashion. Implementations are free to add configuration for local policy and other control of the mechanism, but this is not required.

- REQ 7: The overload control mechanism and any associated default algorithm(s) MUST ensure that the system remains stable. At some point after an overload condition has ended, the mechanism MUST enable capacity to stabilize and become equal to what it would be in the absence of an overload condition. Note that this also requires that the mechanism MUST allow nodes to shed load without introducing non converging oscillations during or after an overload condition.

Compliant. It is possible for an implementation using this to meet this requirement, and the hop-by-hop nature limits the impact of overload control actions. Additional guidance is provided for implementors on sending of the Overload-Metric and its implications for the closed loop control system created by this mechanism.

- REQ 8: Supporting nodes MUST be able to distinguish current overload information from stale information, and SHOULD make decisions using the most currently available information.

Compliant. The mechanism provides for rapid updates of overload control information as well as having timeouts on the validity of overload information that must be provided by senders.

- REQ 9: The mechanism MUST function across fully loaded as well as quiescent transport connections. This is partially derived from the requirement for stability in REQ 7.

Compliant. The mechanism uses piggybacked information transfer, which will generally result in the ability to transfer information on a similar rate to loading. It also provides for triggering the use of DWR with piggybacked information for quiescent connections.

- REQ 10: Consumers of overload information MUST be able to determine when the overload condition improves or ends.

Compliant. The mechanism provides for rapid updates of overload control information, including abatement information, as well as mandatory timeouts on the validity of overload information that must be provided by senders (it is soft state). Additionally, the mechanism provides for sending a DWR with piggybacked information to inform of overload abatement more quickly.

- REQ 11: The overload control mechanism MUST be able to operate in networks of different sizes.

Compliant. The hop-by-hop nature of the mechanism restricts the impacts that large networks might have on the ability of nodes to deal with overload control information, as well as restricting the signaling needed to convey overload information. The use of piggybacked information transfer limits the additional messaging imposed by the mechanism for large and small networks and has the characteristic of scaling with the amount of Diameter traffic on a network. Additionally, the dynamic nature of the capabilities exchange reduces the provisioning burden that can be incurred at large scales.

- REQ 12: When a single network node fails, goes into overload, or suffers from reduced processing capacity, the mechanism MUST make it possible to limit the impact of this on other nodes in the network. This helps to prevent a small-scale failure from becoming a widespread outage.

Compliant. The mechanism provides for information about such issues to be conveyed in order for nodes to take appropriate action to mitigate the situation and prevent cascades.

- REQ 13: The mechanism MUST NOT introduce substantial additional work for node in an overloaded state. For example, a requirement for an overloaded node to send overload information every time it received a new request would introduce substantial work. Existing messaging is likely to have the characteristic of increasing as an overload condition approaches, allowing for the possibility of increased feedback for information piggybacked on it.

Compliant. The mechanism requires sending load and overload information on all messages exchanged with compliant peers. It does not, however, require that the information be recalculated or updated with each message. The update frequency is up to the implementation, and each implementation can make decisions on balancing the update of overload information along with its other priorities. It is expected that using a periodically updated grouped AVP added to all messages sent to compliant peers will not add substantial additional work. Piggyback base transport also does not require composition, sending, or

parsing of new Diameter messages for the purpose of conveying overload control information.

- REQ 14: Some scenarios that result in overload involve a rapid increase of traffic with little time between normal levels and overload inducing levels. The mechanism SHOULD provide for rapid feedback when traffic levels increase.

Compliant. The use of piggybacked information transport by the mechanism allows for overload control information to be sent at the same rate as the normal traffic. It is presumed that the rate of normal traffic will go up as nodes approach, or enter, overload. Additionally, DWR messages may be proactively triggered with piggybacked overload control information to provide overload control information transfer in an ad hoc fashion.

- REQ 15: The mechanism MUST NOT interfere with the congestion control mechanisms of underlying transport protocols. For example, a mechanism that opened additional TCP connections when the network is congested would reduce the effectiveness of the underlying congestion control mechanisms.

Compliant. The mechanism does not require interaction with any underlying congestion control. It relies solely on piggybacked transport and does not request or recommend changes in how the underlying connections are performed.

- REQ 16: The overload control mechanism is likely to be deployed incrementally. The mechanism MUST support a mixed environment where some, but not all, nodes implement it.

Compliant. The mechanism specifies behavior for dealing with non-supporting elements.

- REQ 17: In a mixed environment with nodes that support the overload control mechanism and that do not, the mechanism MUST result in at least as much useful throughput as would have resulted if the mechanism were not present. It SHOULD result in less severe congestion in this environment.

Compliant. When dealing with supporting, and non-supporting nodes, the mechanism specifies behavior that attempts to apply relevant information to decisions on sending to non-compliant hosts. This behavior should result in reductions in traffic that increase the

likelihood of successful overload mitigation in mixed networks.

- REQ 18: In a mixed environment of nodes that support the overload control mechanism and that do not, the mechanism MUST NOT preclude elements that support overload control from treating elements that do not support overload control in an equitable fashion relative to those that do. Users and operators of nodes that do not support the mechanism MUST NOT unfairly benefit from the mechanism. The mechanism specification SHOULD provide guidance to implementors for dealing with elements not supporting overload control.

Compliant. When dealing with supporting, and non-supporting nodes, the mechanism specifies behavior that attempts to apply relevant information to decisions on sending to non-compliant hosts. This allows nodes to treat non-supporting elements in a similar, and fair, fashion relative to non-supporting elements.

- REQ 19: It MUST be possible to use the mechanism between nodes in different realms and in different administrative domains.

Compliant. Scoping of overload information to realms is explicitly specified by the mechanism. There are no requirements imposed by the mechanism that would prevent overload control information from crossing between adjacent nodes that were in separate administrative domains.

- REQ 20: Any explicit overload indication MUST be clearly distinguishable from other errors reported via Diameter.

Compliant. A new grouped AVP conveys all overload control information, and this is transported on existing messages that are not related to overload control. No existing Diameter error codes are used by the mechanism. One new transient error code is defined by the mechanism.

- REQ 21: In cases where a network node fails, is so overloaded that it cannot process messages, or cannot communicate due to a network failure, it may not be able to provide explicit indications of the nature of the failure or its levels of congestion. The mechanism MUST result in at least as much useful throughput as would have resulted if the overload control mechanism was not in place.

Compliant. Procedures are defined cases where supporting nodes become too overloaded to send overload information. No retries or sending of additional messages are required during overload that would reduce useful throughput in these situations.

- REQ 22: The mechanism MUST provide a way for a node to throttle the amount of traffic it receives from a peer node. This throttling SHOULD be graded so that it can be applied gradually as offered load increases. Overload is not a binary state; there may be degrees of overload.

Compliant. The mechanism provides a 32 bit overload severity indication. Interpretation of the value is specific to the algorithm being employed. In the case of the mandatory to implement loss algorithm, the values 0-100 are used to progressively control the amount of traffic dropped.

- REQ 23: The mechanism MUST provide sufficient information to enable a load balancing node to divert messages that are rejected or otherwise throttled by an overloaded upstream node to other upstream nodes that are the most likely to have sufficient capacity to process them.

Compliant. The mechanism provides information so that a load balancing node can determine that an upstream node is in overload. Additionally, it provides load information that can be used as input for balancing decisions.

- REQ 24: The mechanism MUST provide a mechanism for indicating load levels even when not in an overloaded condition, to assist nodes making decisions to prevent overload conditions from occurring.

Compliant. The mechanism provides load information in each message as well as guidelines for implementing the determination of load to be sent.

- REQ 25: The base specification for the overload control mechanism SHOULD offer general guidance on which message types might be desirable to send or process over others during times of overload, based on application-specific considerations. For example, it may be more beneficial to process messages for existing sessions ahead of new sessions. Some networks may have a requirement to give priority to requests associated with emergency sessions. Any normative or otherwise

detailed definition of the relative priorities of message types during an overload condition will be the responsibility of the application specification.

Compliant. Some guidance is provided for priority selection and how to deal with different priority messages is described in an example algorithm implementation.

- REQ 26: The mechanism MUST NOT prevent a node from prioritizing requests based on any local policy, so that certain requests are given preferential treatment, given additional retransmission, not throttled, or processed ahead of others.

Compliant. The mechanism does not place restrictions on how decisions are made to prioritize messages.

- REQ 27: The overload control mechanism MUST NOT provide new vulnerabilities to malicious attack, or increase the severity of any existing vulnerabilities. This includes vulnerabilities to DoS and DDoS attacks as well as replay and man-in-the middle attacks. Note that the Diameter base specification [RFC6733] lacks end to end security and this must be considered.

Compliant. The hop-by-hop nature of the mechanism allows existing Diameter security mechanisms to be used for securing the connections between peers. ***Detailed analysis by persons with security expertise would be beneficial.***

- REQ 28: The mechanism MUST NOT depend on being deployed in environments where all Diameter nodes are completely trusted. It SHOULD operate as effectively as possible in environments where other nodes are malicious; this includes preventing malicious nodes from obtaining more than a fair share of service. Note that this does not imply any responsibility on the mechanism to detect, or take countermeasures against, malicious nodes.

Compliant. Using a hop-by-hop mechanism limits the scope of potentially malicious information. Guidance is provided for trust, in particular relative to scopes. Additional specification around trust relationships could be useful to clarify authorization of overload control information. ***Detailed analysis by persons with security expertise would be beneficial.***

- REQ 29: It MUST be possible for a supporting node to make authorization decisions about what information will be sent to peer nodes based on the identity of those nodes. This allows a domain administrator who considers the load of their nodes to be sensitive information to restrict access to that information. Of course, in such cases, there is no expectation that the overload control mechanism itself will help prevent overload from that peer node.

Compliant. The mechanism provides guidance for authorization decisions and takes no action to restrict local policy when dealing with authorization.
Detailed analysis by persons with security expertise would be beneficial.

- REQ 30: The mechanism MUST NOT interfere with any Diameter compliant method that a node may use to protect itself from overload from non-supporting nodes, or from denial of service attacks.

Compliant. The mechanism allows for local policy overrides for the bulk of its behavior.

- REQ 31: There are multiple situations where a Diameter node may be overloaded for some purposes but not others. For example, this can happen to an agent or server that supports multiple applications, or when a server depends on multiple external resources, some of which may become overloaded while others are fully available. The mechanism MUST allow Diameter nodes to indicate overload with sufficient granularity to allow clients to take action based on the overloaded resources without unreasonably forcing available capacity to go unused. The mechanism MUST support specification of overload information with granularities of at least "Diameter node", "realm", and "Diameter application", and MUST allow extensibility for others to be added in the future.

Compliant. The mechanism allows for flexible specification on the scope that overload control information applies to. It also allows for additional scopes to be specified as extensions.

- REQ 32: The mechanism MUST provide a method for extending the information communicated and the algorithms used for overload control.

Compliant. The mechanism allows for new algorithms to be specified as extensions. It provides an AVP for communicating overload information that can be interpreted differently by different algorithms. It also provides for extension of information transmitted.

REQ 33: The mechanism **MUST** provide a default algorithm that is mandatory to implement.

Compliant. The mechanism specifies the drop algorithm as mandatory to implement.

REQ 34: The mechanism **SHOULD** provide a method for exchanging overload and load information between elements that are connected by intermediaries that do not support the mechanism.

Not Compliant. Additional analysis is needed.

Appendix C. Extending the Overload Mechanism

This specification includes two key extension points to allow for new behaviors to be smoothly added to the mechanism in the future. The following sections discuss the means by which future documents are expected to extend the mechanism.

C.1. New Algorithms

In order to provide the ability for different means of traffic abatement in the future, this specification allows for descriptions of new traffic reduction algorithms. In general, documents that define new algorithms need to describe externally-observable node behavior in sufficient detail as to allow interoperation.

At a minimum, such description needs to include:

1. The name and IANA-registered number for negotiating the algorithm (see Section 5.3).
2. A clear description of how the Overload-Metric AVP is to be interpreted, keeping in mind that "0" is reserved to indicate that no overload condition exists.
3. An example, proof-of-concept description (preferably in pseudo-code) of how nodes can implement the algorithm.

New algorithms must be capable of working with all applications, not just a subset of applications.

It is generally expected that new algorithms will make use of the available overload control information as specified in this document. However, if additional information is needed, the Load-Info AVP allows for additional optional AVPs to be included. It is recommended that designers of any new AVPs defined for this purpose consider reusing existing AVPs first, and also design their AVPS so that they may be reused by others when possible.

C.2. New Scopes

Because it is impossible to foresee all the potential constructs that it might be useful to scope operations to for the purposes of overload, we allow for the registration of new scopes.

At a minimum, such description needs to include:

1. The name and IANA-registered number for negotiating and indicating the scope (see Section 5.4).
2. A syntax for the "Details" field of the Overload-Info-Scope AVP, preferably derived from one of the base Diameter data types.
3. An explicit and unambiguous description of how both parties to the overload control mechanism can determine which transactions correspond to the indicated scope.
4. A clear and exhaustive list that extends the one in Section 2.2, indicating exactly which combinations of scopes are allowed with the new scope. This list must take into account all of the IANA-registered scopes at the time of its publication.

It is acceptable for new scopes to be specific to constructs within one or several applications. In other words, it may be desirable to define scopes that can be applied to one kind of application while not making sense for another. Extension documents should be very clear that such is the case, however, if they choose to do so.

Appendix D. Design Rationale

The current design proposed in this document takes into account several trade-offs and requirements that may not be immediately obvious. The remainder of this appendix highlights some of the potentially more controversial and/or non-obvious of these, and attempts to explain why such decisions were made they way they were.

That said, none of the following text is intended to represent a line in the sand. All of the decisions can be revisited if necessary, especially if additional facts are brought into the analysis that change the balance of the decisions.

D.1. Piggybacking

The decision to piggyback load information on existing messages derives primarily from REQ 14 in [I-D.ietf-dime-overload-reqs]: "The mechanism SHOULD provide for increased feedback when traffic levels increase. The mechanism MUST NOT do this in such a way that it increases the number of messages while at high loads."

If we were to introduce new messaging -- say, by defining a new overload control Application -- then a node in overload would be required to generate more messages at high load in order to keep overload information in its peers up-to-date.

If further analysis determines that other factors are ultimately more important than the provisions of REQ 14, several factors would need to be considered.

First and foremost would be the prohibition, in the base Diameter specification ([RFC6733]), against adding new commands to an existing application. Specifically, section 1.3.4 stipulates: "a new Diameter application MUST be created when one or more of the following criteria are met:... A new command is used within the existing application either because an additional command is added, an existing command has been modified so that a new Command Code had to be registered, or a command has been deleted." Because of this stipulation, the addition of new command codes to existing applications would require registration of entirely new application IDs for those applications to support overload control. We consider this to be too disruptive a change to consider.

By the author's reading, there is no provision that exempts the "Diameter Common Messages" Application (Application ID 0) from the above clauses. This effectively prohibits the addition of new messages to this Application. While it may be theoretically possible to specify behavior that hijacks the DWR/DWA watchdog messages for the purpose of overload control messaging, doing so requires a complete redefinition of their behavior and, fundamentally, their semantics. This approach seems, at first blush, to be an unacceptable change to the base Application.

The remaining approach -- defining a new application for overload control -- has some promise, if we decide not to fulfill REQ 14. It remains to be seen whether the users of the Diameter protocol, including other SDOs who define applications for Diameter, are willing to specify the use of multiple Diameter Applications for use on a single reference point.

D.2. Load AVP in All Packets

Some have questioned the currently specified behavior of message senders including a Load AVP in every message sent. This is being proposed as a potential performance enhancement, with the idea being that message recipients can save processing time by examining arbitrarily selected messages for load information, rather than looking for a Load AVP in every message that arrives. Of course, to enable this kind of sampling, the Load AVP must be guaranteed to be present; otherwise, attempts to find it will occasionally fail.

The reciprocal approach, of sending a Load AVP only when the Load has changed (or changed by more than a certain amount), requires the recipient to search through the Load-Info grouped AVP in every message received in order to determine whether a Load AVP is present.

On a cursory analysis, we determined that appending a Load AVP to each message is fundamentally a cheaper operation than traversing the contents of each Load-Info AVP to determine whether a Load AVP is present.

If a later decision is made to require examination of each message to determine whether it include a Load AVP, we may be able to obtain some efficiencies by requiring Load to be the first AVP in the Load-Info AVP.

D.3. Graceful Failure

Some commenters have raised the question of whether a node can reject an incoming connection upon recognizing that the remote node does not support the Diameter overload control mechanism. One suggestion has been to add a response code to indicate exactly such a situation.

So far, we have opted against doing so. Instead, we anticipate an incremental deployment of the overload control mechanism, which will likely consist of a mixture of nodes that support and node that do not support the mechanism. Were we to allow the rejection of connections that do not support the mechanism, we would create a situation that necessitates a "flag day," on which every Diameter node in a network is required to simultaneously, and in perfect synchronization, switch from not supporting the overload mechanism, to supporting it.

Given the operational difficulty of the foregoing, we have decided that defining a response code, even if optional, that was to be used to reject connections merely for the lack of overload control support, would form an attractive nuisance for implementors. The result could easily be a potential operational nightmare for network

operators.

Authors' Addresses

Adam Roach
Mozilla
Dallas, TX
US

Email: adam@nostrum.com

Eric McMurry
Tekelec
Dallas, TX
US

Email: emcmurphy@computer.org

DIME
Internet-Draft
Intended status: Standards Track
Expires: January 16, 2014

H. Tschofenig, Ed.
Nokia Siemens Networks
J. Korhonen
Renesas Mobile
G. Zorn
Network Zen
K. Pillay
Oracle Communications
July 15, 2013

Diameter AVP Level Security: Scenarios and Requirements
draft-tschofenig-dime-e2e-sec-req-01.txt

Abstract

This specification discusses requirements for providing Diameter security at the level of individual Attribute Value Pairs.

Status of This Memo

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

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

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

This Internet-Draft will expire on January 16, 2014.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of

the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	2
2. Terminology	3
3. Use Case	3
4. Requirements	5
5. Security Considerations	7
6. IANA Considerations	7
7. Acknowledgments	7
8. References	7
8.1. Normative References	7
8.2. Informative References	7
Authors' Addresses	8

1. Introduction

The Diameter Base specification [2] offers security protection between neighboring Diameter peers and mandates that either TLS (for TCP), DTLS (for SCTP), or IPsec is used. These security protocols offer a wide range of security properties, including entity authentication, data-origin authentication, integrity, confidentiality protection and replay protection. They also support a large number of cryptographic algorithms, algorithm negotiation, and different types of credentials.

The need to also offer additional security protection of AVPs between non-neighboring Diameter nodes was recognized very early in the work on Diameter. This lead to work on Diameter security using the Cryptographic Message Syntax (CMS) [3]. Due to lack of deployment interest at that time (and the complexity of the developed solution) the specification was, however, never completed.

In the meanwhile Diameter had received a lot of deployment interest from the cellular operator community and because of the sophistication of those deployments the need for protecting Diameter AVPs between non-neighboring nodes re-surfaced. Since early 2000 (when the work on [3] was discontinued) the Internet community had seen advances in cryptographic algorithms (for example, authenticated encryption algorithms were developed) and new security building blocks were developed.

This document collects requirements for developing a solution to protect Diameter AVPs.

2. Terminology

The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'MAY', and 'OPTIONAL' in this specification are to be interpreted as described in [1].

This document re-uses terminology from the Diameter base specification [2].

3. Use Case

Consider the following use case shown in Figure 1 where a Diameter client wants to interact with its home Diameter server in the example.com realm. The visited domain the Diameter client is attached to makes use of a AAA interconnection provider, shown as AAA Broker in our example. While both the administrators of the visited as well as the home domain are likely to main a business relationship with the intermediate AAA broker network they may want to ensure that certain Diameter AVPs are not sent in the clear or are integrity protected. Note that the security services are likely offered between Diameter Proxy A and Diameter Proxy D for ease of deployment. Proxy A may act on behalf of the Diameter client and Diameter Proxy D acts on behalf of Diameter Server X and Y it serves.

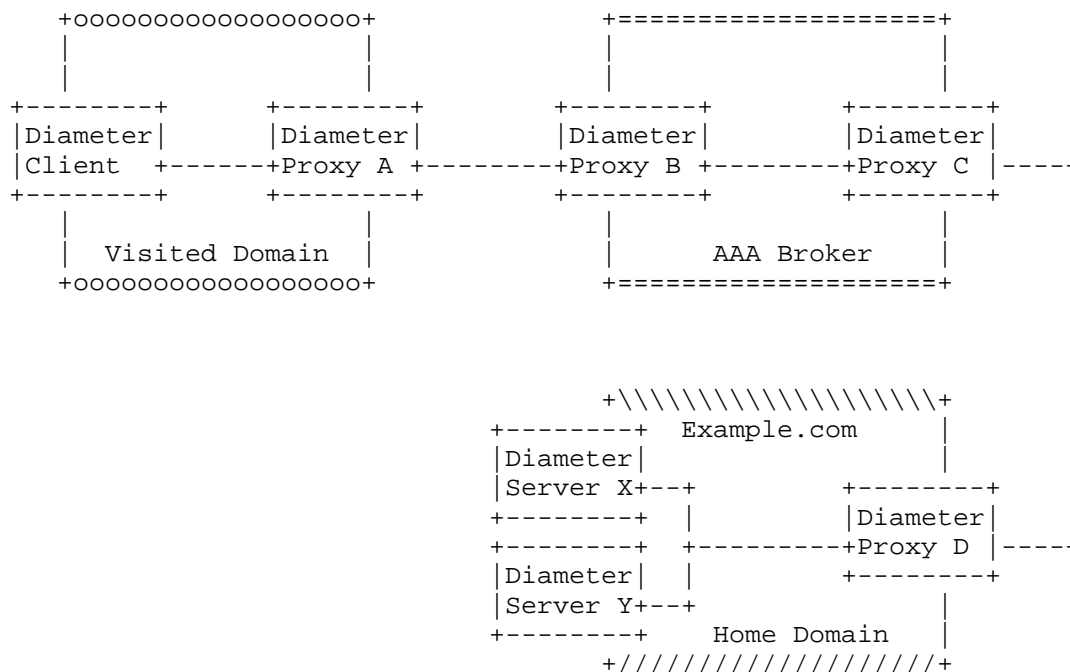


Figure 1: Example Diameter Deployment Setup.

Based on Figure 1 the following use cases can be differentiated. AVP refers to an unprotected AVP and {AVP}k refers to an AVP that experiences security protection without further distinguishing between integrity and confidentiality protection.

In the first scenario, shown in Figure 2, end-to-end security protection is provided between the Diameter client and the Diameter server. Diameter AVPs exchanged between these two Diameter nodes are protected.

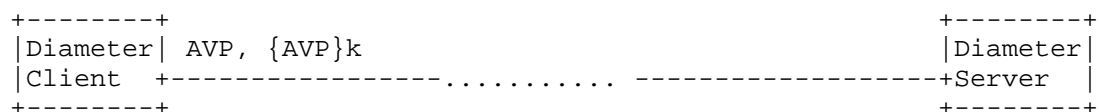


Figure 2: End-to-End Diameter AVP Security Protection.

In the second scenario, shown in Figure 3, a Diameter proxy acts on behalf of the Diameter client with regard to security protection. It applies security protection to outgoing Diameter AVPs and verifies incoming AVPs.

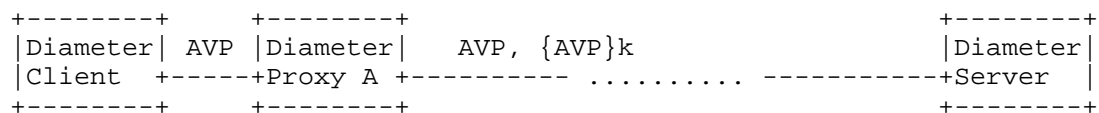


Figure 3: Middle-to-End Diameter AVP Security Protection.

In the third scenario shown in Figure 4 a Diameter proxy acts on behalf of the Diameter server.



Figure 4: End-to-Middle Diameter AVP Security Protection.

The forth and final scenario (see Figure 5) is a combination of the end-to-middle and the middle-to-end scenario shown in Figure 4 and in Figure 3. From a deployment point of view this scenario is easier to accomplish for two reasons: First, Diameter clients and Diameter servers remain unmodified. This ensures that no modifications are needed to the installed Diameter infrastructure. Second, key management is also simplified since fewer number of key pairs need to be negotiated and provisioned.

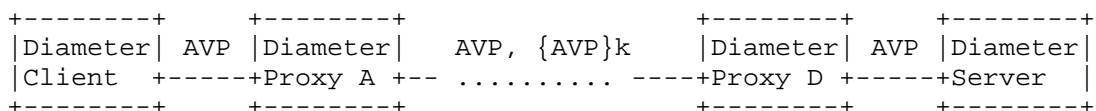


Figure 5: Middle-to-Middle Diameter AVP Security Protection.

Various security threats are mitigated by selectively applying security protection for individual Diameter AVPs. Without protection there is the possibility for password sniffing, confidentiality violation, AVP insertion, deletion or modification. Additionally, applying digital signature offers non-repudiation capabilities; a feature not yet available in today's Diameter deployment. Modification of certain Diameter AVPs may not necessarily be the act of malicious behavior but could also be the result of misconfiguration. An over-aggressively configured firewalling Diameter proxy may also remove certain AVPs. In most cases data origin authentication and integrity protection of AVPs will provide most benefits for existing deployments with minimal overhead and (potentially) operating in a full-backwards compatible manner.

4. Requirements

Requirement #1: Solutions MUST support an extensible set of cryptographic algorithms.

Motivation: Crypto-agility is the ability of a protocol to adapt to evolving cryptographic algorithms and security requirements. This may include the provision of a modular mechanism to allow cryptographic algorithms to be updated without substantial disruption to deployed implementations.

Requirement #2: Solutions MUST support confidentiality, integrity, and data-origin authentication. Solutions for integrity protection MUST work in a backwards-compatible way with existing Diameter applications.

Requirement #3: Solutions MUST support replay protection. Any Diameter node has an access to network time and thus can synchronise their clocks.

Requirement #4: Solutions MUST support the ability to delegate security functionality to another entity

Motivation: As described in Section 3 the ability to let a Diameter proxy to perform security services on behalf of all clients within the same administrative domain is important for incremental deployability. The same applies to the other communication side where a load balancer terminates security services for the servers it interfaces.

Requirement #5: Solutions MUST be able to selectively apply their cryptographic protection to certain Diameter AVPs.

Motivation: Some Diameter applications assume that certain AVPs are added, removed, or modified by intermediaries. As such, it may be necessary to apply security protection selectively.

Requirement #6: Solutions MUST recommend a mandatory-to-implement cryptographic algorithm.

Motivation: For interoperability purposes it is beneficial to have a mandatory-to-implement cryptographic algorithm specified (unless profiles for specific usage environments specify otherwise).

Requirement #7: Solutions MUST support symmetric keys and asymmetric keys.

Motivation: Symmetric and asymmetric cryptographic algorithms provide different security services. Asymmetric algorithms, for example, allow non-repudiation services to be offered.

Requirement #8: A solution for dynamic key management has to be provided. It is assumed that no "new" key management protocol needs to be developed; instead existing ones are re-used, if at all possible. Rekeying could be triggered by (a) management actions and (b) expiring keying material.

Requirement #9: The ability to statically provisioned keys (symmetric as well as asymmetric keys) has to be supported to simplify management for small-scale deployments that typically do not have a backend network management infrastructure.

Requirement #10: Capability/Policy Discovery: This document talks about selectively protecting Diameter AVPs between different Diameter nodes. A Diameter node has to be configured such that it applies security protection to a certain number of AVPs. A number of policy related questions arise: What keying material should be used so that the intended recipient is also able to verify it? What AVPs shall be protected so that the result is not rejected by the recipient? In case of confidentiality protection the Diameter node encrypting AVPs needs to know ahead of time what other node is intended to decrypt them. Should the list of integrity protected AVP be indicated in the protected payload itself (or is it known based on out-of-band information)? Is this policy / capability information assumed to be established out-of-band (manually) or is there a protocol mechanism to distribute this information?

Requirement #11: Command-Line Support: Should solutions allow the provisioning of long-term shared symmetric credentials via a command-line interface / text file? This allows easier management for small-scale deployments.

5. Security Considerations

This entire document focused on the discussion of new functionality for securing Diameter AVPs selectively between non-neighboring nodes.

6. IANA Considerations

This document does not require actions by IANA.

7. Acknowledgments

We would like to thank Guenther Horn for his review comments.

8. References

8.1. Normative References

- [1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [2] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.

8.2. Informative References

- [3] Calhoun, P., Farrell, S., and W. Bulley, "Diameter CMS Security Application", draft-ietf-aaa-diameter-cms-sec-04 (work in progress), March 2002.

Authors' Addresses

Hannes Tschofenig (editor)
Nokia Siemens Networks
Linnoitustie 6
Espoo 02600
Finland

Phone: +358 (50) 4871445
Email: Hannes.Tschofenig@gmx.net
URI: <http://www.tschofenig.priv.at>

Jouni Korhonen
Renesas Mobile
Porkkalankatu 24
Helsinki 00180
Finland

Email: jouni.nospam@gmail.com

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

Email: glenzorn@gmail.com

Kervin Pillay
Oracle Communications
100 Crosby Drive
Bedford, Massachusetts 01730
USA

Email: kervin.pillay@oracle.com

DIME
Internet-Draft
Intended status: Standards Track
Expires: August 22, 2013

H. Tschofenig, Ed.
Nokia Siemens Networks
February 18, 2013

A Keying Database for Diameter End-to-End Security
draft-tschofenig-dime-keying-database-00.txt

Abstract

The Diameter Base specification offers security protection between neighboring Diameter peers using TLS, DTLS, and IPsec. The development of a solution to protect Diameter Attribute Value Pairs between non-neighboring nodes is currently work in progress.

Diameter nodes maintain different types of databases, depending on their functions. Examples include the peer table and the realm-based routing table. This document describes a conceptual model for a keying database as it would be used by a Diameter node to determine what AVPs to protect, and what keys / algorithms to use. On the receiving side it allows the receiving node to select the appropriate security association for verifying the protected AVPs. The design is similar to IPsec and inspired by the routing protocol security key table.

Status of this Memo

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

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

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

This Internet-Draft will expire on August 22, 2013.

Copyright Notice

Copyright (c) 2013 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction	3
2. Terminology	4
3. Conceptual Keying Database	5
4. Examples	8
5. Security Considerations	10
6. IANA Considerations	11
7. Acknowledgments	12
8. References	13
8.1. Normative References	13
8.2. Informative References	13
Author's Address	14

1. Introduction

The Diameter Base specification [RFC6733] offers security protection between neighboring Diameter peers and mandates that either TLS (for TCP), DTLS (for SCTP), or IPsec is used. These security protocols offer a wide range of security properties, including entity authentication, data-origin authentication, integrity, confidentiality protection and replay protection. They also support a large number of cryptographic algorithms, algorithm negotiation, and different types of credentials.

In the meanwhile Diameter had received a lot of deployment interest and the need for protecting Diameter AVPs between non-neighboring nodes has been created. The requirements for Diameter end-to-end security at the level of individual AVPs is provided in [I-D.tschofenig-dime-e2e-sec-req].

This document describes a conceptual model for a keying database as it would be used by a Diameter node to determine what AVPs to protect, what keys and algorithms to use. On the receiving side it allows the receiving node to select the appropriate security association for verifying the protected AVPs. The design is similar to IPsec [RFC4301] and inspired by the routing protocol security key table [I-D.ietf-karp-crypto-key-table].

2. Terminology

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

3. Conceptual Keying Database

Diameter nodes maintain different types of databases, depending on their functions. Examples include the peer table and the realm-based routing table. The usage of the end-to-end security mechanisms described in this document adds another database to those nodes supporting this functionality.

We describe the keying database in a conceptual way as a table, where each row represents a single symmetric or asymmetric key.

The columns that the table consists of are listed as follows:

KeyName: The KeyName is a string identifying the key. On the sender-side it provides information about what key to use for applying integrity and/or confidentiality protection. On the receiver-side this value provides information about the key to use for verification.

DestinationRealm: The DestinationRealm provides information for the sending host to decide what messages to protect. This selector field contains information about the designation realm. The format of a DiameterIdentity. This field may be empty.

DestinationHost: The DestinationHost provides information for the sending host to decide what messages to protect. This selector field contains information about the destination host. The format of a DiameterIdentity. This field may be empty.

ApplicationID: The ApplicationID provides information for the sending host to decide what messages to protect. This field contains a list of comma separated application id values. This field may be empty. The value "*" refers to all application ids that match the DestinationRealm and/or DestinationHost field.

AVPCodeList: The AVPCodeList provides information for the sending host to decide what AVPs need to experience integrity, and optionally confidentiality protection. This selector field contains a list of AVP codes. The value "*" indicates that the protection covers all AVP fields included in the message.

KDF: The KDF field indicates which key derivation function is used to generate short-lived keys from a long-lived symmetric key in the Key field. For symmetric keys, when the long-lived shared key is intended for direct use, the KDF field is set to "none". This document re-used the KDF algorithm registry established in [I-D.ietf-karp-crypto-key-table]. The protocol indicates what (if any) KDFs are valid. For asymmetric algorithm the KDF is left

empty.

AlgID: The AlgID field indicates the cryptographic algorithm used with the security protocol. The algorithm may be an encryption algorithm and mode (such as AES-128-CBC), an authentication algorithm (such as HMAC-SHA1-96 or AES-128-CMAC), or an algorithm applicable to asymmetric cryptography (such as RS256 indicating RSA with SHA-256). If the KDF field contains "none", then a long-lived shared secret key is used directly with this algorithm, otherwise the derived short-lived symmetric key is used with this algorithm. When the long-lived key is used to generate a set of short-lived keys for use with the security protocol, the AlgID field identifies a ciphersuite rather than a single cryptographic algorithm.

KeyType: The KeyType provides information about the type of key found in the Key field. Two values are possible: "SymmetricKey" and "AsymmetricKey".

Key: The Key field contains a symmetric or an asymmetric key. A lower-case hexadecimal string is used for representing a symmetric key. For asymmetric keys the NI URI format [I-D.farrell-decade-ni] is used. The size of the Key field depends on the type of key, the selected KDF, and the AlgID. For instance, a KDF=none and AlgID=AES128 requires a 128-bit symmetric key, which is represented by 32 hexadecimal digits.

Direction: The Direction field indicates whether this key may be used for inbound traffic, outbound traffic, both, or whether the key has been disabled and may not currently be used at all. The supported values are "in", "out", "both", and "disabled", respectively.

SendNotBefore: The NotBefore field specifies the earliest date and time in Universal Coordinated Time (UTC) at which this key should be considered for use. The format is YYYYMMDDHHSSZ, where four digits specify the year, two digits specify the month, two digits specify the day, two digits specify the hour, two digits specify the minute, and two digits specify the second. The "Z" is included as a clear indication that the time is in UTC. This field is empty if the key is for immediate use. If the Direction field indicates that the key is used not used for outbound traffic then this field is ignored.

SendNotAfter: The SendNotAfter field specifies the latest date and time at which this key should be considered for use when sending messages. The format is the same as the SendNotBefore field. If the Direction field indicates that the key is used not used for

outbound traffic then this field is ignored.

RcvNotBefore: The RcvNotBefore field specifies the earliest date and time in Universal Coordinated Time (UTC) at which this key should be considered for use when processing received messages. The format is YYYYMMDDHHSSZ, where four digits specify the year, two digits specify the month, two digits specify the day, two digits specify the hour, two digits specify the minute, and two digits specify the second. The "Z" is included as a clear indication that the time is in UTC. This field is empty if there is no restriction regarding the use of the key when processing received messages. If the Direction field indicates that the key is used not used for inbound traffic then this field is ignored.

RcvNotAfter: The RcvNotAfter field specifies the latest date and time at which this key should be considered for use when processing received traffic. The format of this field is identical to the format of NotBefore. If the Direction field indicates that the key is used not used for inbound traffic then this field is ignored.

KeyManagement: Specifies whether a entry was statically configured or dynamically discovered. This field may help to create a new key when the existing key is expired. An empty field indicates a statically configured key. Values are reserved for automated key management protocols.

4. Examples

This section gives a few examples. For editorial reasons (i.e., the per-line character limit of Internet drafts) a list representation is used instead of a table.

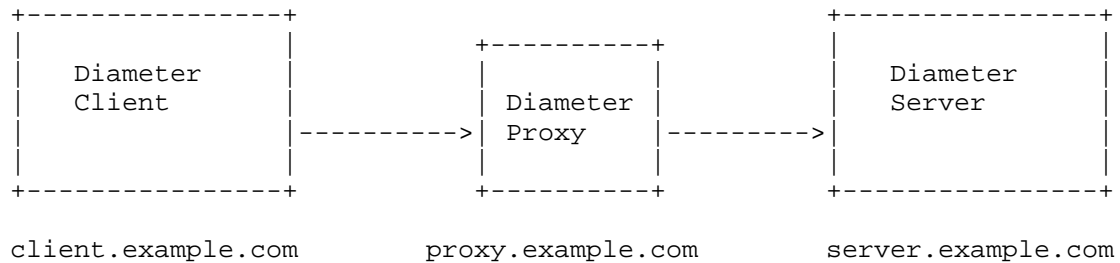


Figure 1: Example Diameter Deployment Setup.

The first example illustrates an entry in the key table at a Diameter client.

```
KeyName: abc123
DestinationRealm:
DestinationHost: server.example.com
ApplicationID: *
AVPCodeList: *
KDF: none
AlgID: HMAC-SHA1-96
KeyType: SymmetricKey
Key: 617CAA833BEF64D88E45
Direction: out
SendNotBefore:
SendNotAfter: 201302142000Z
RcvNotBefore:
RcvNotAfter :
KeyManagement:
```

The second example illustrates the entries of the key database for an asymmetric key as stored at the Diameter client.

KeyName: abc123
DestinationRealm:
DestinationHost: server.example.com
ApplicationID: *
AVPCodeList: *
KDF: none
AlgID: RS256
KeyType: AsymmetricKey
Key: ni:///sha-256;UyaQV-Ev4rdLoHyJJWCi11OHfrYv9ElaGQAlMO2X_-Q
Direction: both
SendNotBefore:
SendNotAfter: 201302142000Z
RcvNotBefore:
RcvNotAfter : 201302142000Z
KeyManagement:

5. Security Considerations

This document focuses on the description of a keying database for usage with Diameter to protect AVPs end-to-end.

It has been recognized in [RFC4107] that automated key management is not viable in multiple scenarios. The conceptual database specified in this document is designed to accommodate both manual key management and automated key management. A future specification to automatically populate rows in the database is envisioned.

Designers should recognize the warning provided in [RFC4107]:

"Automated key management and manual key management provide very different features. In particular, the protocol associated with an automated key management technique will confirm the liveness of the peer, protect against replay, authenticate the source of the short-term session key, associate protocol state information with the short-term session key, and ensure that a fresh short-term session key is generated. Moreover, an automated key management protocol can improve the interoperability by including negotiation mechanisms for cryptographic algorithms. These valuable features are impossible or extremely cumbersome to accomplish with manual key management."

6. IANA Considerations

[Editor's Note: An IANA consideration section will be provided in a future version of this document.]

7. Acknowledgments

I would like to thank the authors of [I-D.ietf-karp-crypto-key-table] for their work. This document is inspired by their writeup.

8. References

8.1. Normative References

- [I-D.farrell-decade-ni]
Farrell, S., Kutscher, D., Dannewitz, C., Ohlman, B., Keraenen, A., and P. Hallam-Baker, "Naming Things with Hashes", draft-farrell-decade-ni-10 (work in progress), August 2012.
- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.
- [RFC6733] Fajardo, V., Arkko, J., Loughney, J., and G. Zorn, "Diameter Base Protocol", RFC 6733, October 2012.

8.2. Informative References

- [I-D.ietf-karp-crypto-key-table]
Housley, R., Polk, T., Hartman, S., and D. Zhang, "Database of Long-Lived Symmetric Cryptographic Keys", draft-ietf-karp-crypto-key-table-05 (work in progress), February 2013.
- [RFC4107] Bellovin, S. and R. Housley, "Guidelines for Cryptographic Key Management", BCP 107, RFC 4107, June 2005.
- [RFC4301] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, December 2005.

Author's Address

Hannes Tschofenig (editor)
Nokia Siemens Networks
Linnoitustie 6
Espoo 02600
Finland

Phone: +358 (50) 4871445
Email: Hannes.Tschofenig@gmx.net
URI: <http://www.tschofenig.priv.at>

