Diameter Maintenance and Extensions
Internet-Draft
Intended status: Standards Track
Expires: January 7, 2017

L. Bertz
Sprint

July 6, 2016

Diameter Domain and Arbitrary Mask Filters
draft-bertz-dime-domainmasks-00

Abstract

This document defines optional Diameter attributes to specify filters by hosts/domains and attributes required by Software Defined Networking (SDN) enabled enforcement points.

Table of Contents

1. Introduction

An optional Diameter Attribute Value Pair (AVP) is defined in this document for DNS support. It is the DNS-Filters AVP which contains a list of DNS query names that follow the registered name QNAME format defined in [RFC1034], Section 3.7.1 and subsequent updates, notably [RFC4592].

DNS Filter AVPs are used in various operator policy enforcement applications such as child protections, parental controls and flow based rating.

Current practices require any updates to these applications to be sideloaded, i.e. updated via a file, on the enforcement points or

proprietary mechanism. As an AVP, the list of entries in the AVP may be dynamically updated per current Diameter application specifications, e.g. [RFC7155].

The second AVP, ArbitraryMask-AVP, defines an arbitrary bitmask for various maskable AVPs used in Classifiers [RFC5777].

Prefix based masks are present in [RFC5777] but the technologies such as Software Defined Networking (SDN) introduce new arbitrary bitmasks on multiple data types in the various protocol headers.

2. Terminology

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

3. DNS-Filters and ArbitraryMasks Attributes

3.1. DNS-Filters AVP

The DNS-Filters AVP (AVP Code TBD1) is of type Grouped and specifies the list of QNAMES [RFC1034] that restrict ip flows (5-tuples) that match the Classifier.

```
 DNS-Filters ::= < AVP Header: TBD1 >
                          1*{ <QNAMES> }
                          * [ AVP ]
```

QNAMES are defined in [RFC1034], Section 3.7.1.

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

3.2. Software Defined Network (SDN) Attributes

3.2.1. IP-Address-Mask-Pattern AVP

The IP-Address-Mask-Pattern AVP (AVP Code TBD2) is of type OctetString. The value is 4 octets specifying the bit positions of a IP address that are taken for matching.

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

3.2.2. Flow-Label AVP

The Flow-Label AVP (AVP Code TBD3) is of type Unsigned32. This field identifies the Flow Label value to be matched.

According to [RFC2460], the flow label is 20 bits long. For the purpose of this specification, the sender of this option MUST prefix the flow label value with 12 bits of "0" before inserting it in the Flow-Label AVP. The receiver SHOULD ignore the first 12 bits of this field before using it in comparisons with flow labels in packets.

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

3.2.3. Flow-Label-Mask-Pattern AVP

The Flow-Label-Mask-Pattern AVP (AVP Code TBD3) is of type OctetString. The value is 4 octets specifying the bit positions of a Flow Label that are taken for matching.

According to [RFC2460], the flow label is 20 bits long. For the purpose of this specification, the sender of this option MUST prefix the flow label value with 12 bits of "0" before inserting it in the Flow Label Mask Pattern AVP. The receiver SHOULD ignore the first 12 bits of this field before using it as a mask.

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

4. Examples

4.1. DNS Filter

The Classifier AVP (AVP Code 511) specified in [RFC5777] is a grouped AVP that consists of a set of attributes that specify how to match a packet. The addition of the DNS-Filter AVP is shown here.

```
Classifier ::= < AVP Header: 511 >
                 { Classifier-ID }
                 [ Protocol ]
                 [ Direction ]
                 [ DNS-Filters ]
               * [ From-Spec ]
               * [ To-Spec ]
               * [ Diffserv-Code-Point ]
                 [ Fragmentation-Flag ]
               * [ IP-Option ]
               * [ TCP-Option ]
                 [ TCP-Flags ]
               * [ ICMP-Type ]
               * [ ETH-Option ]
               * [ AVP ]
```

Setting the DNS-Filter value would specify the capture flows whose DNS related communications match the QUERY values specified in the Classifier.

DNS related communications are those DNS queries which have valid DNS responses for scope implied by the container of the Classifier, i.e. if the Classifer is contained by a Filter-Rule installed during a mobility session then only resulting IP addresses conatined in DNS traffic for the mobility session is used in the selection of flows.

The use of the DNS-Filters AVP implies that the following conditions exist:

- The related DNS traffic is known by the enforcement point
- The related DNS traffic is viewable by the enforcement point

Such conditions are met in network services related to child protections and parental controls.

Use of secured DNS communications [RFC4033] is recommended. In such a case the enforemcent point may be a DNS relay used, as part of the network access service, to enforce or report generate Diameter events in applications such as Application Detection and Charging [TGPPCC] using DNS-Filters.

4.2. SDN Related AVP application

The From-Spec AVP (AVP Code 515) and To-Spec AVP (AVP Code 516) specified in [RFC5777] are grouped AVPs that consist of a set of attributes that specify how to match a packet in the From/To

direction (respectively). The addition of the SDN related AVPs are shown here.

```
From-Spec ::= < AVP Header: 515 >
              * [ IP-Address ]
              * [ IP-Address-Range ]
              * [ IP-Address-Mask ]
              * [ MAC-Address ]
              * [ MAC-Address-Mask]
              * [ EUI64-Address ]
              * [ EUI64-Address-Mask]
              * [ Port ]
              * [ Port-Range ]
                [ Negated ]
                [ Use-Assigned-Address ]
              * [ IP-Address-Mask-Pattern ]
              * [ Flow-Label ]
              * [ Flow-Label-Mask-Pattern ]
              * [ AVP ]

To-Spec ::= < AVP Header: 516 >
            * [ IP-Address ]
            * [ IP-Address-Range ]
            * [ IP-Address-Mask ]
            * [ MAC-Address ]
            * [ MAC-Address-Mask]
            * [ EUI64-Address ]
            * [ EUI64-Address-Mask]
            * [ Port ]
            * [ Port-Range ]
              [ Negated ]
              [ Use-Assigned-Address ]
            * [ IP-Address-Mask-Pattern ]
            * [ Flow-Label ]
            * [ Flow-Label-Mask-Pattern ]
            * [ AVP ]
```

Setting the IP-Address-Mask-Pattern value would permit the capture of an arbitrarily masked IP pattern the flow, e.g. 192.0.22.0 with an arbitraty mask of 255.0.255.0. NOTE that if the value of this AVP is only ’1’ starting from the least significant bit it is no different than the IP-Address-Mask which could be used instead. For example, an arbitrary mask of 255.255.255.0 applied to 192.168.5.0 is no different than the prefix notation 192.168.5.0/24.

Setting the Flow-Label value would permit capture of IPv6 packets with a specific Flow Label. Setting the Flow-Label-Mask-Pattern

permits the capture of flows that match the appropriate mask pattern applied to the Flow Label.

5. IANA Considerations

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	AVP Code	Section Defined	Data Type
DNS-Filters	TBD1	Section 3.1	GROUPED
IP-Address-Mask-Pattern	TBD2	Section 3.2.1	OctetString
Flow-Label	TBD3	Section 3.2.2	Unsigned32
Flow-Label-Mask-Pattern	TBD4	Section 3.2.3	OctetString

6. Security Considerations

6.1. DNS-Filters Usage

The use of DNS-Filters AVP implies visible access to DNS traffic associated with the containing rule’s, e.g. Filter-Rule [RFC5777]. It does not imply a specific design for DNS communications but it is strongly recommended that secured DNS communciations [RFC4033] are used.

In other words, it is NOT recommended to turn off secured DNS communications if this value is present. Rather, a Diameter application appropriate error should be thrown to indicate the rule cannot be enforced.

An operator that can accomplish the same use cases, e.g. child protection or parental control, without the need to see customer DNS traffic is encouraged to do so.

6.2. SDN AVP Usage

This document describes SDN related attributes an extension of [RFC5777].

It introduces a new to/from arbitraty mask for IP addresses. As this is an extension to [RFC5777], which already has an IP-Address-Mask defined it does not raise new security concerns.

This document does introduce the ability to filter upon the IPv6 Flow Labels [RFC2460]. Even though this is a visible IP header it was not previously defined in Diameter related filters but has been used for activities such as Traffic Selection [RFC6088].

Per [RFC2460] the flow label is used by a source to request special handling by routers. If a specific type of host has known flow label usage, i.e. signature, inspection of this field over one or more flows may yield information. However, the flow label is visible in the IPv6 header and thus this information is discernable by any device with access to the IPv6 packet. Recommendations on the security considerations of IPv6 flow usage is outside of the scope of this document.

7. References

7.1. Normative References

[RFC1034] Mockapetris, P., "Domain names - concepts and facilities", STD 13, RFC 1034, DOI 10.17487/RFC1034, November 1987, <http://www.rfc-editor.org/info/rfc1034>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC2460] Deering, S. and R. Hinden, "Internet Protocol, Version 6 (IPv6) Specification", RFC 2460, DOI 10.17487/RFC2460, December 1998, <http://www.rfc-editor.org/info/rfc2460>.

[RFC5777] Korhonen, J., Tschofenig, H., Arumaithurai, M., Jones, M., Ed., and A. Lior, "Traffic Classification and Quality of Service (QoS) Attributes for Diameter", RFC 5777, DOI 10.17487/RFC5777, February 2010, <http://www.rfc-editor.org/info/rfc5777>.

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

[TGPPCC] 3rd Generation Partnership Project, "Policy Charging and Control (PCC); Reference Points, (release 13), 3GPP TS 29.212 v. 13.5.0", 2016-04.

7.2. Informative References

[RFC4033] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "DNS Security Introduction and Requirements", RFC 4033, DOI 10.17487/RFC4033, March 2005, <http://www.rfc-editor.org/info/rfc4033>.

[RFC4592] Lewis, E., "The Role of Wildcards in the Domain Name System", RFC 4592, DOI 10.17487/RFC4592, July 2006, <http://www.rfc-editor.org/info/rfc4592>.

[RFC6088] Tsirtsis, G., Giarreta, G., Soliman, H., and N. Montavont, "Traffic Selectors for Flow Bindings", RFC 6088, DOI 10.17487/RFC6088, January 2011, <http://www.rfc-editor.org/info/rfc6088>.

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

Author’s Address

Lyle Bertz
Sprint
6220 Sprint Parkway
Overland Park, KS 66251
United States

Email: lylebe551144@gmail.com

Diameter Maintenance and Extensions
Internet-Draft
Intended status: Standards Track
Expires: January 7, 2017

L. Bertz
Sprint

July 6, 2016

Diameter Predicted Units
draft-bertz-dime-predictunits-00

Abstract

This document specifies the conveyance of predicted usage information for proper dimensioning of network services that use Diameter based authorization.

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 7, 2017.

Copyright Notice

Table of Contents

1. Introduction

When a User is authorized to use a service via Diameter applications such as [RFC4006] or [RFC7155], the Client is not aware of the average load placed upon it by User. This can lead to overload situations or the Diameter Client redirecting or denying services to valid Users even though their presence may overload the service.

Given virtualization and the use of many software based services the service capacity varies on a service instance, i.e. Diameter Client, basis.

This specification introduces the Predicted-Service-Units Attribute Value Pair (AVP). This information conveys the predicted usage introduced on the service by the authorized User. Such information can be used by the Diameter Client to estimate future load and proactively manage its resources.

Although this informaiton is conveyed from the Diameter Server to the Client several system aspects are out of the scope of this document:

- How the Diameter Server acquired the information contained in the Predicted-Service-UnitsAVP.
- How the values in the Predicted-Service-Units AVP were determined.
- The accuracy or validity of the values in the Predicted-Service-Units AVP.
- Specific actions the Diameter Client should take when its service functions are overloaded.
- Specific actions the Diameter Client takes to bring itself in/out of service for new or existing Users.

2. Terminology

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

3. Predicted Service AVPs

3.1. Predicted-Service-Units

The Predicted-Service-Units AVP (AVP Code TBD1) is of type Grouped and contains the amount of units that the Diameter credit-control client can expect to provide to the end user until the service must be released or the new Credit-Control-Request must be sent, if a Granted-Service-Unit AVP [RFC4006] has been applied to the user's service. A client is not required to implement all of the unit types, and it must ingore unknown or unsupported unit types.

The Predicted-Service-Units AVP is defined as follows (per the grouped- avp-def of [RFC6733]):

```
Predicted-Service-Units ::= < AVP Header: TBD1 >
                             [ CC-Time ]
                             [ CC-Money ]
                             [ CC-Total-Octets ]
                             [ CC-Input-Octets ]
                             [ CC-Output-Octets ]
                             [ CC-Service-Specific-Units ]
                            *[ Service-Cost-Information ]
                            *[ AVP ]
```

With the exception of the Service-Cost-Information AVP, all AVPs are defined in [RFC4006].

The presence of this information is provided as anticipated load information to the Diameter Client and is not intended to be prescriptive in any manner regarding the user's service.

3.2. Service-Cost-Information

The Service-Cost-Information AVP (AVP Code TBD2) is of type Grouped, and it is used to return the service cost information, which the client can use, if it understands it, for various purposes inclduing load prediction and sizing.

A client is not required to implement all of the unit types, and it must ingore unknown or unsupported unit types.

It is defined as follows (per the grouped-avp-def of [RFC6733]):

```
Service-Cost-Information ::= < AVP Header: TBD2 >
                             { Unit-Value }
                             { Cost-Unit }
                             { Service-Dimension }
```

Unit-Value and Cost-Unit are defined in [RFC4006].

3.3. Service-Dimension

The Service-Dimension AVP (AVP Code TBD3) is of type UTF8String. It specifies the applicable service type (dimension) to the Service-Cost-Information.

4. Usage Examples

When Predicted-Service-Units are returned as part of an authorization per [RFC7155] or [RFC4006], the client MAY use this information as guidance on projected load the new user will generate on the service.

If the client supports/understnds the information provided in the Predicted-Service-Units AVP, it can update it's projected load. Based upon this information it MAY take one or more of the following actions (this is not exhaustive):

- Redirect at the service / protocol level any new service requests.
- Begin enforcing mechanisms to reduce the amount of service load on a subset of services already established.

- Remove itself from any system that directs new service requests to it.

- Initiate administrative functions to increase its capacity or start the process of creating new intances to service requests.

5. IANA Considerations

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	AVP Code	Section Defined	Data Type
Predicted-Service-Units	TBD1	Section 3.1	GROUPED
Service-Cost-Information	TBD2	Section 3.2	GROUPED
Service-Dimension	TBD3	Section 3.2	UTF8String

6. Security Considerations

The Diameter base protocol [RFC6733] requires that each Diameter implementation use underlying security; i.e., TLS/TCP, DTLS/SCTP or IPsec. These mechanisms are believed to provide sufficient protection under the normal Internet threat model; that is, assuming that the authorized nodes engaging in the protocol have not been compromised, but that the attacker has complete control over the communication channels between them. This includes eavesdropping, message modification, insertion, and man-in-the-middle and replay attacks. Note also that this application includes a mechanism for application layer replay protection by means of the Session-Id from [RFC6733] and CC- Request-Number, which is specified in this document. The Diameter credit-control application is often used within one domain, and there may be a single hop between the peers. In these environments, the use of TLS/TCP, DTLS/SCTP or IPsec is sufficient. The details of TLS/TCP, DTLS/SCTP or IPsec related security considerations are discussed in the [RFC6733].

Because this application conveys past usage information (directly or indirectly), it increases the interest for various security attacks. Therefore, all parties communicating with each other MUST be authenticated, including, for instance, TLS client-side authentication. In addition, authorization of the client SHOULD be emphasized; i.e., that the client is allowed to perform credit-control for a certain user. The specific means of authorization are

outside of the scope of this specification but can be, for instance, manual configuration.

The attributes provided by this solution MUST be assumed to be privacy sensitive by both the client and server.

7. References

7.1. 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, <http://www.rfc-editor.org/info/rfc2119>.

[RFC4006] Hakala, H., Mattila, L., Koskinen, J-P., Stura, M., and J. Loughney, "Diameter Credit-Control Application", RFC 4006, DOI 10.17487/RFC4006, August 2005, <http://www.rfc-editor.org/info/rfc4006>.

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

7.2. Informative References

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

Author’s Address

Lyle Bertz
Sprint
6220 Sprint Parkway
Overland Park, KS 66251
United States

Email: lylebe551144@gmail.com

Diameter Maintenance and Extensions (DIME)
Internet-Draft
Intended status: Standards Track
Expires: December 23, 2016

S. Donovan
Oracle

June 21, 2016

Diameter Agent Overload and the Peer Overload Report
draft-ietf-dime-agent-overload-06.txt

Abstract

This specification documents an extension to the Diameter Overload Indication Conveyance (DOIC) [RFC7683] base solution. The extension defines the Peer overload report type. The initial use case for the Peer report is the handling of occurrences of overload of a Diameter agent.

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].

Status of This Memo

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

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

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

This Internet-Draft will expire on December 23, 2016.

Copyright Notice

Table of Contents

1. Introduction

This specification documents an extension to the Diameter Overload Indication Conveyance (DOIC) [RFC7683] base solution. The extension defines the Peer overload report type. The initial use case for the Peer report is the handling of occurrences of overload of a Diameter agent.

This document defines the behavior of Diameter nodes when Diameter agents enter an overload condition and send an overload report requesting a reduction of traffic. It also defines new overload report type, the Peer overload report type, that is used for handling of agent overload conditions. The Peer overload report type is defined in a generic fashion so that it can also be used for other Diameter overload scenaios.

The base Diameter overload specification [RFC7683] addresses the handling of overload when a Diameter endpoint (a Diameter Client or Diameter Server as defined in [RFC6733]) becomes overloaded.

In the base specification, the goal is to handle abatement of the overload occurrence as close to the source of the Diameter traffic as is feasible. When possible this is done at the originator of the traffic, generally referred to as a Diameter Client. A Diameter Agent might also handle the overload mitigation. For instance, a Diameter Agent might handle Diameter overload mitigation when it knows that a Diameter Client does not support the DOIC extension.

This document extends the base Diameter endpoint overload specification to address the case when Diameter Agents become overloaded. Just as is the case with other Diameter nodes -- Diameter Clients and Diameter Servers -- surges in Diameter traffic can cause a Diameter Agent to be asked to handle more Diameter traffic than it was configured to handle. For a more detailed discussion of what can cause the overload of Diameter nodes, refer to the Diameter Overload Requirements [RFC7068].

This document defines a new overload report type to communicate occurrences of agent overload. This report type works for the "Loss" overload mitigation algorithm defined in [RFC7683] and is expected to work for other overload abatement algorithms defined in extensions to the DOIC solution.

2. Terminology and Abbreviations

Diameter Node

A RFC6733 Diameter Client, an RFC6733 Diameter Server, and RFC6733 Diameter Agent.

Diameter Endpoint

An RFC6733 Diameter Client and RFC6733 Diameter Server.

Reporting Node

A DOIC Node that sends an overload report in a Diameter answer message.

Reacting Node

A DOIC Node that receives and acts on a DOIC overload report.

DOIC Node

A Diameter Node that supports the DOIC solution defined in [RFC7683].

3. Peer Report Use Cases

This section outlines representative use cases for the peer report used to communicate agent overload.

There are two primary classes of use cases currently identified, those involving the overload of agents and those involving overload of Diameter endpoints. In both cases the goal is to use an overload algorithm that controls traffic sent towards peers.

3.1. Diameter Agent Overload Use Cases

The peer report needs to support the following use cases.

3.1.1. Single Agent

This use case is illustrated in Figure 1. In this case, the client sends all traffic through the single agent. If there is a failure in the agent then the client is unable to send Diameter traffic toward the server.

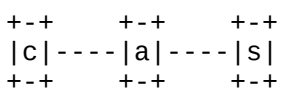

Figure 1

A more likely case for the use of agents is illustrated in Figure 2. In this case, there are multiple servers behind the single agent. The client sends all traffic through the agent and the agent determines how to distribute the traffic to the servers based on local routing and load distribution policy.

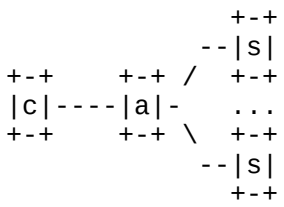

Figure 2

In both of these cases, the occurrence of overload in the single agent must by handled by the client in a similar fashion as if the client were handling the overload of a directly connected server. When the agent becomes overloaded it will insert an overload report in answer messages flowing to the client. This overload report will contain a requested reduction in the amount of traffic sent to the agent. The client will apply overload abatement behavior as defined in the base Diameter overload specification [RFC7683] or the extension draft that defines the indicated overload abatement algorithm. This will result in the throttling of the abated traffic that would have been sent to the agent, as there is no alternative route. An appropriate error response is sent back to the originator of the request.

3.1.2. Redundant Agents

Figure 3 and Figure 4 illustrate a second, and more likely, type of deployment scenario involving agents. In both of these cases, the client has Diameter connections to two agents.

Figure 3 illustrates a client that has a primary connection to one of the agents (agent a1) and a secondary connection to the other agent (agent a2). In this scenario, under normal circumstances, the client will use the primary connection for all traffic. The secondary connection is used when there is a failure scenario of some sort.

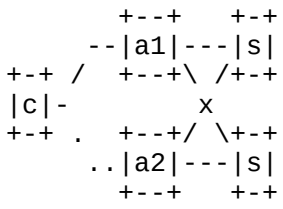

Figure 3

The second case, in Figure 4, illustrates the case where the connections to the agents are both actively used. In this case, the client will have local distribution policy to determine the traffic sent through each client.

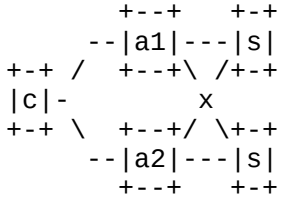

Figure 4

In the case where one of the agents in the above scenario becomes overloaded, the client should reduce the amount of traffic sent to the overloaded agent by the amount requested. This traffic should instead be routed through the non-overloaded agent. For example, assume that the overloaded agent requests a reduction of 10 percent. The client should send 10 percent of the traffic that would have been routed to the overloaded agent through the non-overloaded agent.

When the client has an active and a standby connection to the two agents then an alternative strategy for responding to an overload report from an agent is to change to standby connection to active and route all traffic through the new active connection.

In the case where both agents are reporting overload, the client may need to start decreasing the total traffic sent to the agents. This would be done in a similar fashion as discussed in Section 3.1.1 The amount of traffic depends on the combined reduction requested by the two agents.

3.1.3. Agent Chains

There are also deployment scenarios where there can be multiple Diameter Agents between Diameter Clients and Diameter Servers. An example of this type of deployment include when there are edge agents between Diameter networks.

Figure 5 illustrates one such network deployment case. Note that while this figure shows a maximum of two agents being involved in a Diameter transaction, it is possible that more than two agents could be in the path of a transaction.

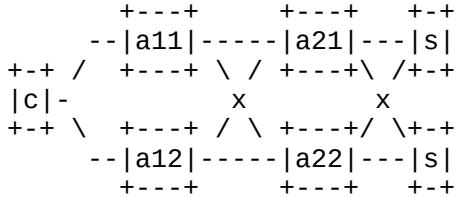

Figure 5

Handling of overload of one or both of agents a11 or a12 in this case is equivalent to that discussed in section 2.2.

Overload of agents a21 and a22 must be handled by the previous hop agents. As such, agents a11 and a12 must handle the overload mitigation logic when receiving an agent overload report from agents a21 and a22.

The handling of peer overload reports is similar to that discussed in Section 3.1.2. If the overload can be addressed using diversion then this approach should be taken.

If both of the agents have requested a reduction in traffic then the previous hop agent must start throttling the appropriate number of transactions. When throttling requests, an agent uses the same error responses as defined in the base DOIC specification [RFC7683].

3.2. Diameter Endpoint Use Cases

This section outlines use cases for the peer overload report involving Diameter Clients and Diameter Servers.

3.2.1. Hop-by-hop Abatement Algorithms

It is envisioned that abatement algorithms will be defined that will support the option for Diameter Endpoints to send peer reports. For instance, it is envisioned that one usage scenario for the rate algorithm, [I-D.ietf-dime-doic-rate-control], which is being worked on by the DIME working group as this document is being written, will involve abatement being done on a hop-by-hop basis.

This rate deployment scenario would involve Diameter Endpoints generating peer reports and selecting the rate algorithm for abatement of overload conditions.

4. Interaction Between Host/Realm and Peer Overload Reports

It is possible that both an agent and an end-point in the path of a transaction are overloaded at the same time. When this occurs, Diameter entities need to handle both overload reports. In this scenario the reacting node should first handle the throttling of the overloaded host or realm. Any messages that survive throttling due to host or realm reports should then go through abatement for the peer overload report. In this scenario, when doing abatement on the PEER report, the reacting node SHOULD take into consideration the number of messages already throttled by the handling of the HOST/ REALM report abatement.

> Note: The goal is to avoid traffic oscillations that might result from throttling of messages for both the HOST/REALM overload reports and the PEER overload reports. This is especially a concern if both reports are of type LOSS.

5. Peer Report Behavior

This section defines the normative behavior associated with the Peer Report extension to the DOIC solution.

5.1. Capability Announcement

5.1.1. Reacting Node Behavior

When sending a Diameter request a DOIC node that supports the OC_PEER_REPORT feature MUST include an OC-Supported-Features AVP with an OC-Feature-Vector AVP with the OC_PEER_REPORT bit set.

When sending a request a DOIC node that supports the OC_PEER_REPORT feature MUST include a SourceID AVP in the OC-Supported-Features AVP with its own DiameterIdentity.

Note: This allows the DOIC nodes in the path of the request to determine if the indication of support came from a Diameter peer or if the request traversed a node that does not support the OC_PEER_REPORT feature.

When an agent relays a request that includes a SourceID AVP in the OC-Supported-Features AVP, a DOIC node that supports the OC_PEER_REPORT feature MUST remove the received SourceID AVP and replace it with a SourceID AVP containing its own Diameter identity.

5.1.2. Reporting Node Behavior

When receiving a request a DOIC node that supports the OC_PEER_REPORT feature MUST update transaction state with an indication of whether or not the peer from which the request was received supports the OC_PEER_REPORT feature.

Note: The transaction state is used when the DOIC node is acting as a peer-report reporting node and needs send OC-OLR reports of type PEER_REPORT in answer messages. The peer overload reports are only included in answer messages being sent to peers that support the OC_PEER_REPORT feature.

The peer supports the OC_PEER_REPORT feature if the received request contains an OC-Supported-Features AVP with the OC-Feature-Vector with the OC_PEER_REPORT feature bit set and with a SourceID AVP with a Diameter ID that matches the DiameterIdentity of the peer from which the request was received.

When an agent relays an answer message, a reporting node that supports the OC_PEER_REPORT feature MUST strip any SourceID AVP from the OC-Supported-Features AVP.

When sending an answer message, a reporting node that supports the OC_PEER_REPORT feature MUST determine if the peer to which the answer is to be sent supports the OC_PEER_REPORT feature.

If the peer supports the OC_PEER_REPORT feature then the reporting node MUST indicate support for the feature in the OC-Supported-Features AVP.

If the peer supports the OC_PEER_REPORT feature then the reporting node MUST insert the SourceID AVP in the OC-Supported-Features AVP in the answer message.

If the peer supports the OC_PEER_REPORT feature then the reporting node MUST insert the OC-Peer-Algo AVP in the OC-Supported-Features AVP. The OC-Peer-Algo AVP MUST indicate the overload abatement

algorithm that the reporting node wants the reacting nodes to use should the reporting node send a peer overload report as a result of becoming overloaded.

5.2. Peer Overload Report Handling

This section defines the behavior for the handling of overload reports of type peer.

5.2.1. Overload Control State

This section describes the Overload Control State (OCS) that might be maintained by both the peer report reporting node and the peer report reacting node.

This is an extension of the OCS handling defined in [RFC7683].

5.2.1.1. Reporting Node Peer Report OCS

A DOIC Node that supports the OC_PEER_REPORT feature SHOULD maintain Reporting Node OCS, as defined in [RFC7683] and extended here.

If different abatement specific contents are sent to each peer then the reporting node MUST maintain a separate reporting node peer report OCS entry per peer to which a peer overload report is sent.

Note: The rate overload abatement algorithm allows for different rates to be sent to each peer.

5.2.1.2. Reacting Node Peer Report OCS

In addition to OCS maintained as defined in [RFC7683], a reacting node that supports the OC_PEER_REPORT feature maintains the following OCS per supported Diameter application:

A peer-type OCS entry for each peer to which it sends requests.

A peer-type OCS entry is identified by the pair of Application-ID and the peer’s DiameterIdentity.

The peer-type OCS entry include the following information (the actual information stored is an implementation decision):

Sequence number (as received in the OC-OLR AVP).

Time of expiry (derived from OC-Validity-Duration AVP received in the OC-OLR AVP and time of reception of the message carrying OC-OLR AVP).

Selected abatement algorithm (as received in the OC-Supported-Features AVP).

Input data that is abatement algorithm specific (as received in the OC-OLR AVP -- for example, OC-Reduction-Percentage for the loss abatement algorithm).

5.2.2. Reporting Node Maintenance of Peer Report OCS

All rules for managing the reporting node OCS entries defined in [RFC7683] apply to the peer report.

5.2.3. Reacting Node Maintenance of Peer Report OCS

When a reacting node receives an OC-OLR AVP with a report type of peer it MUST determine if the report was generated by the Diameter peer from which the report was received.

If a reacting node receives an OC-OLR AVP of type peer and the SourceID matches the ID of the Diameter peer from which the request was received then the report was received from a Diameter peer.

If a reacting node receives an OC-OLR AVP of type peer and the SourceID does not match the ID of the Diameter peer from which the request was received then the reacting node MUST ignore the overload report.

If the Peer Report OLR was received from a Diameter peer then the reacting node MUST determine if it is for an existing or new overload condition.

The OLR is for an existing overload condition if the reacting node has an OCS that matches the received OLR. For a peer report-type, this means it matches the Application-ID and the peer's DiameterIdentity in an existing OCS entry.

If the OLR is for an existing overload condition then it MUST determine if the OLR is a retransmission or an update to the existing OLR.

If the sequence number for the received OLR is greater than the sequence number stored in the matching OCS entry then the reacting node MUST update the matching OCS entry.

If the sequence number for the received OLR is less than or equal to the sequence number in the matching OCS entry then the reacting node MUST silently ignore the received OLR. The matching OCS MUST NOT be updated in this case.

If the received OLR is for a new overload condition then the reacting node MUST generate a new OCS entry for the overload condition.

For a peer report this means it creates an OCS entry with an DiameterID from the SourceID AVP in the received OC-OLR AVP.

If the received OLR contains a validity duration of zero ("0") then the reacting node MUST update the OCS entry as being expired.

The reacting node does not delete an OCS when receiving an answer message that does not contain an OC-OLR AVP (i.e. absence of OLR means "no change").

The reacting node sets the abatement algorithm based on the OC-Peer-Algo AVP in the received OC-Supported-Features AVP.

5.2.4. Peer Report Reporting Node Behavior

When there is an existing reporting node peer report OCS entry, the reporting node MUST include an OC-OLR AVP with a report type of peer using the contents of the reporting node peer report OCS entry in all answer messages sent by the reporting node to peers that support the OC_PEER_REPORT feature.

> The reporting node determines if a peer supports the OC_PEER_REPORT feature based on the indication recorded in the reporting node’s transaction state.

The reporting node MUST include its DiameterIdentity in the SourceID AVP in the OC-OLR AVP. This is used by DOIC nodes that support the OC_PEER_REPORT feature to determine if the report was received from a Diameter peer.

The reporting agent must follow all other overload reporting node behaviors outlined in the DOIC specification.

5.2.5. Peer Report Reacting Node Behavior

A reacting node supporting this extension MUST support the receipt of multiple overload reports in a single message. The message might include a host overload report, a realm overload report and/or a peer overload report.

When a reacting node sends a request it MUST determine if that request matches an active OCS.

In all cases, if the reacting node is an agent then it MUST strip the Peer Report OC-OLR AVP from the message.

If the request matches an active OCS then the reacting node MUST apply abatement treatment on the request. The abatement treatment applied depends on the abatement algorithm indicated in the OCS.

For peer overload reports, the preferred abatement treatment is diversion. As such, the reacting node SHOULD attempt to divert requests identified as needing abatement to other peers.

If there is not sufficient capacity to divert abated traffic then the reacting node MUST throttle the necessary requests to fit within the available capacity of the peers able to handle the requests.

If the abatement treatment results in throttling of the request and if the reacting node is an agent then the agent MUST send an appropriate error as defined in [RFC7683].

In the case that the OCS entry validity duration expires or has a validity duration of zero ("0"), meaning that if the reporting node has explicitly signaled the end of the overload condition then abatement associated with the overload abatement MUST be ended in a controlled fashion.

6. Peer Report AVPs

6.1. OC-Supported-Features AVP

This extension adds a new feature to the OC-Feature-Vector AVP. This feature indication shows support for handling of peer overload reports. Peer overload reports are used by agents to indicate the need for overload abatement handling by the agent’s peer.

A supporting node must also include the SourceID AVP in the OC-Supported-Features capability AVP.

This AVP contains the Diameter Identity of the node that supports the OC_PEER_REPORT feature. This AVP is used to determine if support for the peer overload report is in an adjacent node. The value of this AVP should be the same Diameter identity used as part of the CER/CEA base Diameter capabilities exchange.

This extension also adds the OC-Peer-Algo AVP to the OC-Supported-Features AVP. This AVP is used by a reporting node to indicate the abatement algorithm it will use for peer overload reports.

```
OC-Supported-Features ::= < AVP Header: 621 >
                          [ OC-Feature-Vector ]
                          [ SourceID ]
                          [ OC-Peer-Algo]
                        * [ AVP ]
```

6.1.1. OC-Feature-Vector

The peer report feature defines a new feature bit is added for the OC-Feature-Vector AVP.

OC_PEER_REPORT (0x0000000000000010)

When this flag is set by a DOIC node it indicates that the DOIC node supports the peer overload report type.

6.1.2. OC-Peer-Algo

The OC-Peer-Algo AVP (AVP code TBD1) is of type Unsigned64 and contains a 64 bit flags field of announced capabilities of a DOIC node. The value of zero (0) is reserved.

Feature bits defined for the OC-Feature-Vector AVP and associated with overload abatement algorithms are reused for this AVP.

6.2. OC-OLR AVP

This extension makes no changes to the SequenceNumber or ValidityDuration AVPs in the OC-OLR AVP. These AVPs are also be used in peer overload reports.

The OC_PEER_REPORT feature extends the base Diameter overload specification by defining a new overload report type of "peer". See section [7.6] in [RFC7683] for a description of the OC-Report-Type AVP.

The overload report MUST also include the Diameter identity of the agent that generated the report. This is necessary to handle the case where there is a non supporting agent between the reporting node and the reacting node. Without the indication of the agent that generated the overload request, the reacting node could erroneously assume that the report applied to the non-supporting node. This could, in turn, result in unnecessary traffic being either redistributed or throttled.

The SourceID AVP is used in the OC-OLR AVP to carry this DiameterIdentity.

```
OC-OLR ::= < AVP Header: 623 >
           < OC-Sequence-Number >
           < OC-Report-Type >
           [ OC-Reduction-Percentage ]
           [ OC-Validity-Duration ]
           [ SourceID ]
         * [ AVP ]
```

6.2.1. OC-Report-Type AVP

The following new report type is defined for the OC-Report-Type AVP.

PEER_REPORT 2 The overload treatment should apply to all requests bound for the peer identified in the overload report. If the peer identified in the overload report is not a peer to the reacting endpoint then the overload report should be stripped and not acted upon.

6.3. SourceID

The SourceID AVP (AVP code TBD2) is of type DiameterIdentity and is inserted by a Diameter node to indicate the source of the AVP in which it is a part.

In the case of peer reports, the SourceID AVP indicates the node that supports this feature (in the OC-Supported-Features AVP) or the node that generates an overload with a report type of peer (in the OC-OLR AVP).

It contains the DiameterIdentity of the inserting node. This is used by other Diameter nodes to determine the node that inserted the enclosing AVP that contains the SourceID AVP.

6.4. Attribute Value Pair flag rules

Attribute Name	AVP Code	Section Defined	Value Type	AVP flag rules: MUST	AVP flag rules: MUST NOT
OC-Peer-Algo	TBD1	x.x	Unsigned64		V
SourceID	TBD2	x.x	DiameterIdentity		V

7. IANA Considerations

7.1. AVP codes

New AVPs defined by this specification are listed in Section 6. All AVP codes are allocated from the 'Authentication, Authorization, and Accounting (AAA) Parameters' AVP Codes registry.

7.2. New registries

There are no new IANA registries introduced by this document.

The values used for the OC-Peer-Algo AVP are the subset of the "OC-Feature-Vector AVP Values (code 622)" registry. Only the values in that registry that apply to overload abatement algorithms apply to the OC-Peer-Algo AVP.

8. Security Considerations

Agent overload is an extension to the base Diameter overload mechanism. As such, all of the security considerations outlined in [RFC7683] apply to the agent overload scenarios.

It is possible that the malicious insertion of an agent overload report could have a bigger impact on a Diameter network as agents can be concentration points in a Diameter network. Where an end-point report would impact the traffic sent to a single Diameter server, for example, a peer report could throttle all traffic to the Diameter network.

This impact is amplified in an agent that sits at the edge of a Diameter network that serves as the entry point from all other Diameter networks.

9. Acknowledgements

Adam Roach and Eric McMurry for the work done in defining a comprehensive Diameter overload solution in draft-roach-dime-overload-ctrl-03.txt.

Ben Campbell for his insights and review of early versions of this document.

10. Normative References

[I-D.ietf-dime-doic-rate-control]
Donovan, S. and E. Noel, "Diameter Overload Rate Control", draft-ietf-dime-doic-rate-control-03 (work in progress), March 2016.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

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

[RFC7068] McMurry, E. and B. Campbell, "Diameter Overload Control Requirements", RFC 7068, DOI 10.17487/RFC7068, November 2013, <http://www.rfc-editor.org/info/rfc7068>.

[RFC7683] Korhonen, J., Ed., Donovan, S., Ed., Campbell, B., and L. Morand, "Diameter Overload Indication Conveyance", RFC 7683, DOI 10.17487/RFC7683, October 2015, <http://www.rfc-editor.org/info/rfc7683>.

Author’s Address

Steve Donovan
Oracle
7460 Warren Parkway, Suite 300
Frisco, Texas 75034
United States

Email: srdonovan@usdonovans.com

Diameter Maintenance and Extensions (DIME)
Internet-Draft
Intended status: Standards Track
Expires: April 7, 2017

S. Donovan, Ed.
Oracle
E. Noel
AT&T Labs

October 4, 2016

Diameter Overload Rate Control
draft-ietf-dime-doic-rate-control-04.txt

Abstract

This specification documents an extension to the Diameter Overload Indication Conveyance (DOIC) [RFC7683] base solution. This extension adds a new overload control abatement algorithm. This abatement algorithm allows for a DOIC reporting node to specify a maximum rate at which a DOIC reacting node sends Diameter requests to the DOIC reporting node.

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].

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 7, 2017.

Copyright Notice

Table of Contents

1. Introduction

This document defines a new Diameter overload control abatement algorithm.

The base Diameter overload specification [RFC7683] defines the loss algorithm as the default Diameter overload abatement algorithm. The loss algorithm allows a reporting node to instruct a reacting node to reduce the amount of traffic sent to the reporting node by abating (diverting or throttling) a percentage of requests sent to the server. While this can effectively decrease the load handled by the server, it does not directly address cases where the rate of arrival of service requests increases quickly. If the service requests that result in Diameter transactions increases quickly then the loss algorithm cannot guarantee the load presented to the server remains below a specific rate level. The loss algorithm can be slow to protect the stability of reporting nodes when subjected with rapidly changing loads.

Consider the case where a reacting node is handling 100 service requests per second, where each of these service requests results in one Diameter transaction being sent to a reacting node. If the reacting node is approaching an overload state, or is already in an overload state, it will send a Diameter overload report requesting a percentage reduction in traffic sent. Assume for this discussion that the reporting node requests a 10% reduction. The reacting node will then abate (diverting or throttling) ten Diameter transactions a second, sending the remaining 90 transactions per second to the reacting node.

Now assume that the reacting node’s service requests spikes to 1000 requests per second. The reacting node will continue to honor the reporting nodes request for a 10% reduction in traffic. This results, in this example, in the reacting node sending 900 Diameter transactions per second, abating the remaining 100 transactions per second. This spike in traffic is significantly higher than the reporting node is expecting to handle and can result in negative impacts to the stability of the reporting node.

The reporting node can, and likely would, send another overload report requesting that the reacting node abate 91% of requests to get back to the desired 90 transactions per second. However, once the spike has abated and the reacting node handled service requests returns to 100 per second, this will result in just 9 transactions per second being sent to the reporting node, requiring a new overload report setting the reduction percentage back to 10%. This control feedback loop has the potential to make the situation worse.

One of the benefits of a rate based algorithm is that it better handles spikes in traffic. Instead of sending a request to reduce traffic by a percentage, the rate approach allows the reporting node to specify the maximum number of Diameter requests per second that can be sent to the reporting node. For instance, in this example, the reporting node could send a rate-based request specifying the maximum transactions per second to be 90. The reacting node will send the 90 regardless of whether it is receiving 100 or 1000 service requests per second.

This document extends the base DOIC solution [RFC7683] to add support for the rate based overload abatement algorithm.

This document draws heavily on work in the SIP Overload Control working group. The definition of the rate abatement algorithm is copied almost verbatim from the SOC document [RFC7415], with changes focused on making the wording consistent with the DOIC solution and the Diameter protocol.

2. Terminology and Abbreviations

Diameter Node

A RFC6733 Diameter Client, RFC6733 Diameter Server, or RFC6733 Diameter Agent.

Diameter Endpoint

An RFC6733 Diameter Client or RFC6733 Diameter Server.

DOIC Node

A Diameter Node that supports the DOIC solution defined in [RFC7683].

Reporting Node

A DOIC Node that sends a DOIC overload report.

Reacting Node

A DOIC Node that receives and acts on a DOIC overload report.

3. Interaction with DOIC report types

As of the publication of this specification there are two DOIC report types defined with the specification of a third in progress:

1. Host - Overload of a specific Diameter Application at a specific Diameter Node as defined in [RFC7683].

2. Realm - Overload of a specific Diameter Application at a specific Diameter Realm as defined in [RFC7683].

3. Peer - Overload of a specific Diameter peer as defined in [I-D.ietf-dime-agent-overload].

The rate algorithm MAY be selected by reporting nodes for any of these report types.

It is expected that all report types defined in the future will indicate whether or not the rate algorithm can be used with that report type.

4. Capability Announcement

This extension defines the rate abatement algorithm (referred to as rate in this document) feature. Support for the rate feature will be reflected by use of a new value, as defined in Section 6.1.1, in the OC-Feature-Vector AVP per the rules defined in [RFC7683].

Note that Diameter nodes that support the rate feature will, by definition, support both the loss and rate based abatement algorithms. DOIC reacting nodes SHOULD indicate support for both the loss and rate algorithms in the OC-Feature-Vector AVP.

> There may be local policy reasons that cause a DOIC node that supports the rate abatement algorithm to not include it in the OC-Feature-Vector. All reacting nodes, however, must continue to include loss in the OC-Feature-Vector in order to remain compliant with [RFC7683].

A reporting node MAY select one abatement algorithm to apply to host and realm reports and a different algorithm to apply to peer reports.

> For host or realm reports the selected algorithm is reflected in the OC-Feature-Vector AVP sent as part of the OC-Supported-Features AVP included in answer messages for transaction where the request contained an OC-Supported-Features AVP. This is per the procedures defined in [RFC7683].
>
> For peer reports the selected algorithm is reflected in the OC-Peer-Algo AVP sent as part of the OC-Supported-Features AVP included answer messages for transactions where the request contained an OC-Supported-Features AVP. This is per the procedures defined in [I-D.ietf-dime-agent-overload].

Editor's Node: The peer report specification is still under development and, as such, the above paragraph is subject to change.

5. Overload Report Handling

This section describes any changes to the behavior defined in [RFC7683] for handling of overload reports when the rate overload abatement algorithm is used.

5.1. Reporting Node Overload Control State

A reporting node that uses the rate abatement algorithm SHOULD maintain reporting node Overload Control State (OCS) for each reacting node to which it sends a rate Overload Report (OLR).

This is different from the behavior defined in [RFC7683] where a single loss percentage sent to all reacting nodes.

A reporting node SHOULD maintain OCS entries when using the rate abatement algorithm per supported Diameter application, per targeted reacting node and per report-type.

A rate OCS entry is identified by the tuple of Application-Id, report-type and DiameterID of the target of the rate OLR.

A reporting node that supports the rate abatement algorithm MUST include the rate of its abatement algorithm in the OC-Maximum-Rate AVP when sending a rate OLR.

All other elements for the OCS defined in [RFC7683] and [I-D.ietf-dime-agent-overload] also apply to the reporting nodes OCS when using the rate abatement algorithm.

5.2. Reacting Node Overload Control State

A reacting node that supports the rate abatement algorithm MUST indicate rate as the selected abatement algorithm in the reacting node OCS when receiving a rate OLR.

A reacting node that supports the rate abatement algorithm MUST include the rate specified in the OC-Maximum-Rate AVP included in the OC-OLR AVP as an element of the abatement algorithm specific portion of reacting node OCS entries.

All other elements for the OCS defined in [RFC7683] and [I-D.ietf-dime-agent-overload] also apply to the reporting nodes OCS when using the rate abatement algorithm.

5.3. Reporting Node Maintenance of Overload Control State

A reporting node that has selected the rate overload abatement algorithm and enters an overload condition MUST indicate rate as the abatement algorithm in the resulting reporting node OCS entries.

A reporting node that has selected the rate abatement algorithm and enters an overload condition MUST indicate the selected rate in the resulting reporting node OCS entries.

When selecting the rate algorithm in the response to a request that contained an OC-Supporting-Features AVP with an OC-Feature-Vector AVP indicating support for the rate feature, a reporting node MUST ensure that a reporting node OCS entry exists for the target of the overload report. The target is defined as follows:

- For Host reports the target is the DiameterIdentity contained in the Origin-Host AVP received in the request.
- For Realm reports the target is the DiameterIdentity contained in the Origin-Realm AVP received in the request.
- For Peer reports the target is the DiameterIdentity of the Diameter Peer from which the request was received.

5.4. Reacting Node Maintenance of Overload Control State

When receiving an answer message indicating that the reporting node has selected the rate algorithm, a reacting node MUST indicate the rate abatement algorithm in the reacting node OCS entry for the reporting node.

A reacting node receiving an overload report for the rate abatement algorithm MUST save the rate received in the OC-Maximum-Rate AVP contained in the OC-OLR AVP in the reacting node OCS entry.

5.5. Reporting Node Behavior for Rate Abatement Algorithm

When in an overload condition with rate selected as the overload abatement algorithm and when handling a request that contained an OC-Supported-Features AVP that indicated support for the rate abatement algorithm, a reporting node SHOULD include an OC-OLR AVP for the rate algorithm using the parameters stored in the reporting node OCS for the target of the overload report.

When sending an overload report for the Rate algorithm, the OC-Maximum-Rate AVP is included and the OC-Reduction-Percentage AVP is not included.

5.6. Reacting Node Behavior for Rate Abatement Algorithm

When determining if abatement treatment should be applied to a request being sent to a reporting node that has selected the rate overload abatement algorithm, the reacting node MAY use the algorithm detailed in Section 7.

> Note: Other algorithms for controlling the rate can be implemented by the reacting node as long as they result in the correct rate of traffic being sent to the reporting node.

Once a determination is made by the reacting node that an individual Diameter request is to be subjected to abatement treatment then the procedures for throttling and diversion defined in [RFC7683] and [I-D.ietf-dime-agent-overload] apply.

6. Rate Abatement Algorithm AVPs

6.1. OC-Supported-Features AVP

The rate algorithm does not add any new AVPs to the OC-Supported-Features AVP.

The rate algorithm does add a new feature bit to be carried in the OC-Feature-Vector AVP.

6.1.1. OC-Feature-Vector AVP

This extension adds the following capabilities to the OC-Feature-Vector AVP.

OLR_RATE_ALGORITHM (0x0000000000000004)

> When this flag is set by the overload control endpoint it indicates that the DOIC Node supports the rate overload control algorithm.

6.2. OC-OLR AVP

This extension defines the OC-Maximum-Rate AVP to be an optional part of the OC-OLR AVP.

```
OC-OLR ::= < AVP Header: TBD2 >
           < OC-Sequence-Number >
           < OC-Report-Type >
           [ OC-Reduction-Percentage ]
           [ OC-Validity-Duration ]
           [ SourceID ]
           [ OC-Maximum-Rate ]
         * [ AVP ]
```

This extension makes no changes to the other AVPs that are part of the OC-OLR AVP.

This extension does not define new overload report types. The existing report types of host and realm defined in [RFC7683] apply to the rate control algorithm. The peer report type defined in [I-D.ietf-dime-agent-overload] also applies to the rate control algorithm.

6.2.1. OC-Maximum-Rate AVP

The OC-Maximum-Rate AVP (AVP code TBD1) is type of Unsigned32 and describes the maximum rate that that the sender is requested to send traffic. This is specified in terms of requests per second.

A value of zero indicates that no traffic is to be sent.

6.3. Attribute Value Pair flag rules

Attribute Name	AVP Code	Section Defined	Value Type	AVP flag rules	
				MUST	MUST NOT
OC-Maximum-Rate	TBD1	6.2	Unsigned32		V

7. Rate Based Abatement Algorithm

This section is pulled from [RFC7415], with minor changes needed to make it apply to the Diameter protocol.

7.1. Overview

The reporting node is the one protected by the overload control algorithm defined here. The reacting node is the one that abates traffic towards the server.

Following the procedures defined in [draft-ietf-dime-doic], the reacting node and reporting node signal one another support for rate-based overload control.

Then periodically, the reporting node relies on internal measurements (e.g. CPU utilization or queuing delay) to evaluate its overload state and estimate a target maximum Diameter request rate in number of requests per second (as opposed to target percent reduction in the case of loss-based abatement).

When in an overloaded state, the reporting node uses the OC-OLR AVP to inform reacting nodes of its overload state and of the target Diameter request rate.

Upon receiving the overload report with a target maximum Diameter request rate, each reacting node applies abatement treatment for new Diameter requests towards the reporting node.

7.2. Reporting Node Behavior

The actual algorithm used by the reporting node to determine its overload state and estimate a target maximum Diameter request rate is beyond the scope of this document.

However, the reporting node MUST periodically evaluate its overload state and estimate a target Diameter request rate beyond which it would become overloaded. The reporting node must allocate a portion of the target Diameter request rate to each of its reacting nodes. The reporting node may set the same rate for every reacting node, or may set different rates for different reacting node.

The maximum rate determined by the reporting node for a reacting node applies to the entire stream of Diameter requests, even though abatement may only affect a particular subset of the requests, since the reacting node might apply priority as part of its decision of which requests to abate.

When setting the maximum rate for a particular reacting node, the reporting node may need take into account the workload (e.g. CPU load per request) of the distribution of message types from that reacting node. Furthermore, because the reacting node may prioritize the specific types of messages it sends while under overload

restriction, this distribution of message types may be different from the message distribution for that reacting node under non-overload conditions (e.g., either higher or lower CPU load).

Note that the AVP for the rate algorithm is an upper bound (in request messages per second) on the traffic sent by the reacting node to the reporting node. The reacting node may send traffic at a rate significantly lower than the upper bound, for a variety of reasons.

In other words, when multiple reacting nodes are being controlled by an overloaded reporting node, at any given time some reacting nodes may receive requests at a rate below its target maximum Diameter request rate while others above that target rate. But the resulting request rate presented to the overloaded reporting node will converge towards the target Diameter request rate.

Upon detection of overload, and the determination to invoke overload controls, the reporting node MUST follow the specifications in [RFC7683] to notify its clients of the allocated target maximum Diameter request rate and to notify them that the rate overload abatement is in effect.

The reporting node MUST use the OC-Maximum-Rate AVP defined in this specification to communicate a target maximum Diameter request rate to each of its clients.

7.3. Reacting Node Behavior

7.3.1. Default algorithm

In determining whether or not to transmit a specific message, the reacting node can use any algorithm that limits the message rate to the OC-Maximum-Rate AVP value in units of messages per second. For ease of discussion, we define T = 1/[OC-Maximum-Rate] as the target inter-Diameter request interval. It may be strictly deterministic, or it may be probabilistic. It may, or may not, have a tolerance factor, to allow for short bursts, as long as the long term rate remains below 1/T.

The algorithm may have provisions for prioritizing traffic.

If the algorithm requires other parameters (in addition to "T", which is 1/OC-Maximum-Rate), they may be set autonomously by the reacting node, or they may be negotiated independently between reacting node and reporting node.

In either case, the coordination is out of scope for this document. The default algorithms presented here (one with and one without provisions for prioritizing traffic) are only examples.

To apply abatement treatment to new Diameter requests at the rate specified in the OC-Maximum-Rate AVP value sent by the reporting node to its reacting nodes, the reacting node MAY use the proposed default algorithm for rate-based control or any other equivalent algorithm that forward messages in conformance with the upper bound of 1/T messages per second.

The default Leaky Bucket algorithm presented here is based on [ITU-T Rec. I.371] Appendix A.2. The algorithm makes it possible for reacting nodes to deliver Diameter requests at a rate specified in the OC-Maximum-Rate value with tolerance parameter TAU (preferably configurable).

Conceptually, the Leaky Bucket algorithm can be viewed as a finite capacity bucket whose real-valued content drains out at a continuous rate of 1 unit of content per time unit and whose content increases by the increment T for each forwarded Diameter request. T is computed as the inverse of the rate specified in the OC-Maximum-Rate AVP value, namely T = 1 / OC-Maximum-Rate.

Note that when the OC-Maximum-Rate value is 0 with a non-zero OC-Validity-Duration, then the reacting node should apply abatement treatment to 100% of Diameter requests destined to the overloaded reporting node. However, when the OC-Validity-Duration value is 0, the reacting node should stop applying abatement treatment.

If, at a new Diameter request arrival, the content of the bucket is less than or equal to the limit value TAU, then the Diameter request is forwarded to the server; otherwise, the abatement treatment is applied to the Diameter request.

Note that the capacity of the bucket (the upper bound of the counter) is (T + TAU).

The tolerance parameter TAU determines how close the long-term admitted rate is to an ideal control that would admit all Diameter requests for arrival rates less than 1/T and then admit Diameter requests precisely at the rate of 1/T for arrival rates above 1/T. In particular at mean arrival rates close to 1/T, it determines the tolerance to deviation of the inter-arrival time from T (the larger TAU the more tolerance to deviations from the inter-departure interval T).

This deviation from the inter-departure interval influences the admitted rate burstyness, or the number of consecutive Diameter requests forwarded to the reporting node (burst size proportional to TAU over the difference between 1/T and the arrival rate).

In situations where reacting nodes are configured with some knowledge about the reporting node (e.g., operator pre-provisioning), it can be beneficial to choose a value of TAU based on how many reacting nodes will be sending requests to the reporting node.

Reporting nodes with a very large number of reacting nodes, each with a relatively small arrival rate, will generally benefit from a smaller value for TAU in order to limit queuing (and hence response times) at the reporting node when subjected to a sudden surge of traffic from all reacting nodes. Conversely, a reporting node with a relatively small number of reacting nodes, each with proportionally larger arrival rate, will benefit from a larger value of TAU.

Once the control has been activated, at the arrival time of the k-th new Diameter request, ta(k), the content of the bucket is provisionally updated to the value

X' = X - (ta(k) - LCT)

where X is the value of the leaky bucket counter after arrival of the last forwarded Diameter request, and LCT is the time at which the last Diameter request was forwarded.

If X' is less than or equal to the limit value TAU, then the new Diameter request is forwarded and the leaky bucket counter X is set to X' (or to 0 if X' is negative) plus the increment T, and LCT is set to the current time ta(k). If X' is greater than the limit value TAU, then the abatement treatment is applied to the new Diameter request and the values of X and LCT are unchanged.

When the first response from the reporting node has been received indicating control activation (OC-Validity-Duration>0), LCT is set to the time of activation, and the leaky bucket counter is initialized to the parameter TAU0 (preferably configurable) which is 0 or larger but less than or equal to TAU.

TAU can assume any positive real number value and is not necessarily bounded by T.

TAU=4*T is a reasonable compromise between burst size and abatement rate adaptation at low offered rate.

Note that specification of a value for TAU, and any communication or coordination between servers, is beyond the scope of this document.

A reference algorithm is shown below.

No priority case:

```
// T: inter-transmission interval, set to 1 / OC-Maximum-Rate
// TAU: tolerance parameter
// ta: arrival time of the most recent arrival
// LCT: arrival time of last SIP request that was sent to the server
//      (initialized to the first arrival time)
// X: current value of the leaky bucket counter (initialized to
//    TAU0)

// After most recent arrival, calculate auxiliary variable Xp
Xp = X - (ta - LCT);

if (Xp <= TAU) {
  // Transmit SIP request
  // Update X and LCT
  X = max (0, Xp) + T;
  LCT = ta;
} else {
  // Reject SIP request
  // Do not update X and LCT
}
```

7.3.2. Priority treatment

The reacting node is responsible for applying message priority and for maintaining two categories of requests: Request candidates for reduction, requests not subject to reduction (except under extenuating circumstances when there aren’t any messages in the first category that can be reduced).

Accordingly, the proposed Leaky bucket implementation is modified to support priority using two thresholds for Diameter requests in the set of request candidates for reduction. With two priorities, the proposed Leaky bucket requires two thresholds TAU1 < TAU2:

- All new requests would be admitted when the leaky bucket counter is at or below TAU1,
- Only higher priority requests would be admitted when the leaky bucket counter is between TAU1 and TAU2,

- All requests would be rejected when the bucket counter is above TAU2.

This can be generalized to n priorities using n thresholds for n>2 in the obvious way.

With a priority scheme that relies on two tolerance parameters (TAU2 influences the priority traffic, TAU1 influences the non-priority traffic), always set TAU1 <= TAU2 (TAU is replaced by TAU1 and TAU2). Setting both tolerance parameters to the same value is equivalent to having no priority. TAU1 influences the admitted rate the same way as TAU does when no priority is set. And the larger the difference between TAU1 and TAU2, the closer the control is to strict priority queuing.

TAU1 and TAU2 can assume any positive real number value and is not necessarily bounded by T.

Reasonable values for TAU0, TAU1 & TAU2 are:

- TAU0 = 0,
- TAU1 = 1/2 * TAU2, and
- TAU2 = 10 * T.

Note that specification of a value for TAU1 and TAU2, and any communication or coordination between servers, is beyond the scope of this document.

A reference algorithm is shown below.

Priority case:

```
// T: inter-transmission interval, set to 1 / OC-Maximum-Rate
// TAU1: tolerance parameter of no priority Diameter requests
// TAU2: tolerance parameter of priority Diameter requests
// ta: arrival time of the most recent arrival
// LCT: arrival time of last Diameter request that was sent to the server
//      (initialized to the first arrival time)
// X: current value of the leaky bucket counter (initialized to
//    TAU0)

// After most recent arrival, calculate auxiliary variable Xp
Xp = X - (ta - LCT);

if (AnyRequestReceived && Xp <= TAU1) || (PriorityRequestReceived &&
Xp <= TAU2 && Xp > TAU1) {
  // Transmit Diameter request
  // Update X and LCT
  X = max (0, Xp) + T;
  LCT = ta;
} else {
  // Apply abatement treatment to Diameter request
  // Do not update X and LCT
}
```

7.3.3. Optional enhancement: avoidance of resonance

As the number of reacting node sources of traffic increases and the throughput of the reporting node decreases, the maximum rate admitted by each reacting node needs to decrease, and therefore the value of T becomes larger. Under some circumstances, e.g. if the traffic arises very quickly simultaneously at many sources, the occupancies of each bucket can become synchronized, resulting in the admissions from each source being close in time and batched or very ’peaky’ arrivals at the reporting node, which not only gives rise to control instability, but also very poor delays and even lost messages. An appropriate term for this is ’resonance’ [Erramilli].

If the network topology is such that resonance can occur, then a simple way to avoid resonance is to randomize the bucket occupancy at two appropriate points -- at the activation of control and whenever the bucket empties -- as described below.

After updating the value of the leaky bucket to X’, generate a value u as follows:

if X’ > 0, then u=0

else if X’ <= 0, then let u be set to a random value uniformly distributed between -1/2 and +1/2

Then (only) if the arrival is admitted, increase the bucket by an amount T + uT, which will therefore be just T if the bucket hadn't emptied, or lie between T/2 and 3T/2 if it had.

This randomization should also be done when control is activated, i.e. instead of simply initializing the leaky bucket counter to TAU0, initialize it to TAU0 + uT, where u is uniformly distributed as above. Since activation would have been a result of response to a request sent by the reacting node, the second term in this expression can be interpreted as being the bucket increment following that admission.

This method has the following characteristics:

- If TAU0 is chosen to be equal to TAU and all sources activate control at the same time due to an extremely high request rate, then the time until the first request admitted by each reacting node would be uniformly distributed over [0,T];
- The maximum occupancy is TAU + (3/2)T, rather than TAU + T without randomization;
- For the special case of 'classic gapping' where TAU=0, then the minimum time between admissions is uniformly distributed over [T/2, 3T/2], and the mean time between admissions is the same, i.e. T+1/R where R is the request arrival rate.
- At high load randomization rarely occurs, so there is no loss of precision of the admitted rate, even though the randomized 'phasing' of the buckets remains.

8. IANA Consideration

8.1. AVP codes

New AVPs defined by this specification are listed in Section 6. All AVP codes are allocated from the 'Authentication, Authorization, and Accounting (AAA) Parameters' AVP Codes registry.

8.2. New registries

There are no new IANA registries introduced by this document.

9. Security Considerations

The rate overload abatement mechanism is an extension to the base Diameter overload mechanism. As such, all of the security

considerations outlined in [RFC7683] apply to the rate overload abatement mechanism.

10. Acknowledgements

11. References

11.1. Normative References

[I-D.ietf-dime-agent-overload]
Donovan, S., "Diameter Agent Overload", draft-ietf-dime-agent-overload-00 (work in progress), December 2014.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

[RFC5226] Narten, T. and H. Alvestrand, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 5226, DOI 10.17487/RFC5226, May 2008, <http://www.rfc-editor.org/info/rfc5226>.

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

[RFC7683] Korhonen, J., Ed., Donovan, S., Ed., Campbell, B., and L. Morand, "Diameter Overload Indication Conveyance", RFC 7683, DOI 10.17487/RFC7683, October 2015, <http://www.rfc-editor.org/info/rfc7683>.

11.2. Informative References

[Erramilli]
Erramilli, A. and L. Forys, "Traffic Synchronization Effects In Teletraffic Systems", 1991.

[RFC7415] Noel, E. and P. Williams, "Session Initiation Protocol (SIP) Rate Control", RFC 7415, DOI 10.17487/RFC7415, February 2015, <http://www.rfc-editor.org/info/rfc7415>.

Authors' Addresses

Steve Donovan (editor)
Oracle
17210 Campbell Road
Dallas, Texas 75254
United States

Email: srdonovan@usdonovans.com

Eric Noel
AT&T Labs
200s Laurel Avenue
Middletown, NJ 07747
United States

Email: ecnoel@research.att.com

Diameter Maintenance and Extensions (DIME)
Internet-Draft
Intended status: Standards Track
Expires: December 5, 2016

S. Donovan
Oracle

June 3, 2016

Diameter Routing Message Priority
draft-ietf-dime-drmp-07.txt

Abstract

When making routing and resource allocation decisions, Diameter nodes currently have no generic mechanism to determine the relative priority of Diameter messages. This document addresses this by defining a mechanism to allow Diameter endpoints to indicate the relative priority of Diameter transactions. With this information Diameter nodes can factor that priority into routing, resource allocation and overload abatement decisions.

Status of This Memo

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

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

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

This Internet-Draft will expire on December 5, 2016.

Copyright Notice

Table of Contents

1. Introduction

The Diameter Overload Indication Conveyance (DOIC) solution [RFC7683] for Diameter overload control introduces scenarios where Diameter routing decisions made by Diameter nodes can be influenced by the overload state of other Diameter nodes. This includes the scenarios where Diameter endpoints and Diameter agents can throttle requests as a result of the target for the request being overloaded.

With currently available mechanisms these Diameter nodes do not have a mechanism to differentiate request message priorities when making these throttling decisions. As such, all requests are treated the

same, meaning that all requests have the same probability of being throttled.

There are scenarios where treating all requests the same can cause issues. For instance, it might be considered important to reduce the probability of transactions involving first responders being throttled during overload scenarios caused, for example, by a period of heavy signaling resulting from a natural disaster.

This document defines a mechanism that allows Diameter nodes to indicate the relative priority of Diameter transactions. With this information other Diameter nodes can factor the relative priority of requests into routing and throttling decisions.

1.1. Applicability

There are two primary considerations that must be addressed for the mechanism described in this document to work effectively. The first takes into consideration that the Diameter base protocol defined in [RFC6733] is designed to transport multiple Diameter applications and that Diameter nodes can be implemented that support multiple applications. In order for the DRMP mechanism to work, the priorities defined for all messages across all applications used in a Diameter administrative domain must be defined in a consistent and coordinated fashion, taking the default priority into account. See Section 10 for a discussion of some of the considerations that need to be factored into the setting of DRMP priorities used by Diameter applications.

> Note that this consideration does not apply to Diameter networks where all Diameter nodes only support a single application.

Without this cross application priority design taken into consideration it is possible for messages for one application to gain unwarranted preferential treatment over messages for other applications.

This mechanism also depends on all of the messages that carry the DRMP AVP are inserted into Diameter messages by trusted nodes within the Diameter administrative domain. As discussed in Section 12, misbehaving nodes have the ability to use the DRMP mechanism to gain unwarranted preferential treatment.

When messages cross Diameter administrative boundaries, care should be taken to either strip or modify the DRMP priority values in these messages. If the priority definitions vary between the two Diameter administrative domains then it is possible for messages from a foreign domain to gain unwarranted preferential treatment.

2. Terminology and Abbreviations

Diversion

As defined in [RFC7683]. An overload abatement treatment where the reacting node selects alternate destinations or paths for requests.

DOIC

Diameter Overload Indication Conveyance.

DRMP

Diameter Routing Message Priority.

Overload Abatement

As defined in [RFC7683]. Reaction to receipt of an overload report resulting in a reduction in traffic sent to the reporting node. Abatement actions include diversion and throttling.

Priority

The relative importance of a Diameter message. A lower priority value implies a higher relative importance of the message.

Throttling

As defined in [RFC7683]. An abatement treatment that limits the number of requests sent by the DOIC reacting node. Throttling can include a Diameter Client choosing to not send requests, or a Diameter Agent or Server rejecting requests with appropriate error responses. In both cases the result of the throttling is a permanent rejection of the transaction.

3. Conventions Used in This Document

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

RFC 2119 [RFC2119] interpretation does not apply for the above listed words when they are not used in all-caps format.

4. Problem Statement

With the introduction of overload control mechanisms, Diameter nodes will be required to make decisions regarding which Diameter request messages should be throttled as a result of overloaded Diameter nodes.

There is currently no generic mechanism to indicate which request messages should be given preferential treatment when these throttling decisions are made.

As a result, all messages are treated equally and, as such, have an equal probability of being throttled.

There are a number of scenarios where it is appropriate for an application to mark a request as being of a higher priority than other application requests. These are discussed in the next section.

This document defines a mechanism for applications to indicate priority for individual transactions, reducing the probability of those transactions being throttled if there are other lower priority transactions that are eligible for throttling treatment.

While the primary usage of DRMP defined priorities is for input to Diameter overload control related throttling decisions, it is also expected that the priority information could also be used for other routing related functionality. This might include giving higher priority transactions preferential treatment when selecting routes.

It is also envisioned that DRMP priority information could be used by Diameter endpoints to make resource allocation decisions. For instance, a Diameter Server might choose to use the priority information to treat higher priority requests ahead of lower priority requests. It might also use the priority information to as a reason to fail a request as a result of insufficient resources.

> Note: There are a number of application specific definitions indicating various views of application level priority for different requests. Using these application specific priority Attribute Value Pairs (AVPs) as input to throttling and other Diameter routing decisions would require Diameter agents to understand all applications and do application specific parsing of all messages in order to determine the priority of individual messages. This is considered an unacceptable level of complexity to put on elements whose primary responsibility is to route Diameter messages.

5. Use Cases

This section discusses various scenarios where Diameter transactions can benefit from the use of priority information.

It is important to note that for priority information to be reliably usable, the Diameter nodes sending and consuming DRMP AVPs must have pre-established trust relationships of the sort described in Section 12.

5.1. First Responder Related Signaling

Natural disasters can result in a considerable increase in usage of network resources. This can be made worse if the disaster results in a loss of network capacity.

The combination of added load and reduced capacity can lead to Diameter nodes becoming overloaded and, as a result, the use of DOIC mechanisms to request a reduction in traffic. This in turn results in requests being throttled in an attempt to control the overload scenario and prevent the overloaded node from failing.

There is the need for first responders and other individuals responsible for handling the after effects of the disaster to be assured that they can gain access to the network resources in order to communicate both between themselves and with other network resources.

Signaling associated with first responders needs to be given a higher priority to help ensure they can most effectively do their jobs.

The United States Wireless Priority Services (WPS) and Government Emergency Telecommunications Service (GETS) are examples of systems designed to address the command and control aspects of these first responder needs.

5.2. Emergency Call Related Signaling

Similar to the first responder scenario, there is also signaling associated with emergency calls. Given the critical nature of these emergency calls, this signaling should also be given preferential treatment when possible.

5.3. Differentiated Services

Operators may desire to differentiate network-based services by providing a service level agreement that includes preferential

Diameter routing behavior. This might, for example, be modeled as Platinum, Gold and Silver levels of service.

In this scenario an operator might offer a Platinum SLA that includes ensuring that all signaling for a customer who purchases the Platinum service being marked as having a higher priority than signaling associated with Gold and Silver customers.

5.4. Application Specific Priorities

There are scenarios within Diameter applications where it might be appropriate to give a subset of the transactions for the application a higher priority than other transactions for that application.

For instance, when there is a series of transactions required for a user to gain access to network services, it might be appropriate to mark transactions that occur later in the series at a higher priority than those that occur early in the series. This would recognize that there was potentially significant work done by the network already that would be lost if those later transactions were throttled.

There are also scenarios where an agent cannot easily differentiate a request that starts a session from requests that update or end sessions. In these scenarios it might be appropriate to mark the requests that establish new sessions with a lower priority than updates and session ending requests. This also recognizes that more work has already taken place for established sessions and, as a result, it might be more harmful from a signaling point of view if the session update and session ending requests were to be throttled.

There are also scenarios where the priority of requests for individual command codes within an application depends on the context that exists when the request is sent. There isn’t always information in the message from which this context can be determined by Diameter nodes other than the node that originates the request.

This is similar to the scenario where a series of requests are needed to access a network service. It is different in that the series of requests involve different application command codes. In this scenario requests with the same command code have different implied priorities.

> One example of this is in the 3GPP application [S6a] where an Update Location Request (ULR) request resulting from an MME (Mobility Management Entity) restoration procedure might be given a higher priority than a ULR (Update Location Request) resulting from an initial attach.

6. Theory of Operation

This section outlines the envisioned usage of DRMP.

The expected behavior depends on the role (request sender, agent or request handler) of the Diameter node handling the request.

The following behavior is expected during the flow of a Diameter transaction.

1. Request sender - The sender of a request, be it a Diameter Client or a Diameter Server, determines the relative priority of the request and includes that priority information in the request. The method for determining the relative priority is application specific and is outside the scope of this specification. The request sender also saves the priority information with the transaction state. This will be used when handling the answer messages.

2. Agents handling the request - Agents use the priority information when making routing decisions. This can include determining which requests to route first, which requests to throttle and where the request is routed. For instance, requests with higher priority might have a lower probability of being throttled. The mechanism for how the agent determines which requests are candidates to be throttled is implementation dependent and is outside the scope of this document. Before forwarding request messages, agents generally do not modify the priority information present in the received request message nor include the priority information when absent in the received request message. However, in some scenarios, agents can modify the priority information, for example, edge agents modifying the priority values set by an adjacent operator. There might be other scenarios where a Diameter endpoint does not support the DRMP mechanism and agents insert the priority information in the request messages for that non supporting endpoint. When forwarding the request messages, the agent also saves the transaction priority in the transaction state, either as locally managed state or using the Proxy-Info mechanism defined in [RFC6733]. This will be used when handling the associated answer message for the transaction.

3. Request handler - The handler of the request, be it a Diameter Server or a Diameter Client, can use the priority information to determine how to handle the request. This could include determining the order in which requests are handled and resources that are applied to handling of the request.

4. Answer sender - The handler of the request is also the sender of the answer. The answer sender uses the priority information received in the request message when sending the answer. This implies that answers for higher priority transactions are given preferential treatment to lower priority transactions. The answer sender also has the option of including priority information in the answer message. This is done when the answer message needs to have a different priority than the priority carried in the request message. The priority included by the answer sender is application specific.

5. Agent handling the answer - By default, agents handling answer messages use the priority information stored with the transaction state to determine the priority of relaying the answer message. However, priority information included in the answer message, when present, is used in place of the stored priority information. The use of priority information implies that answers for higher priority transactions are given preferential treatment to lower priority transactions. When forwarding the answer messages, agents generally do not modify the priority information present in the received answer messages nor include the priority information when absent in the received answer messages. However, in some scenarios, agents can modify the priority information, for example, edge agents modifying the priority values set by an adjacent operator. There might be other scenarios where a Diameter endpoint does not support the DRMP mechanism and agents insert the priority information for that non-supporting endpoint.

6. Answer handler - The answer handler uses the same method as the agent to determine the priority of the answer message. By default the handler of the answer message uses the priority saved in the transaction’s state. Priority information in the answer message is used when present. The priority is used when allocating resources for processing that occurs after the receipt of the answer message.

7. Extensibility

This document does not define extensibility mechanisms that are specific to the DRMP mechanism. As a result, any extension that requires new AVPs will be required to use existing Diameter extensibility mechanisms defined in [RFC6733].

8. Normative Behavior

This section contains the normative behavior associated with Diameter Resource Message Priority (DRMP).

When routing priority information is available, Diameter nodes SHOULD include Diameter routing message priority in the DRMP AVP in all Diameter request messages.

> Note: The method of determining the priority value included in the request is application specific and is not in the scope of this specification.

The priority marking scheme does not require the Diameter Agents to understand application specific AVPs.

When available, Diameter nodes SHOULD use routing priority information included in the DRMP AVP when making Diameter overload throttling decisions.

Diameter agents MAY use routing priority information included in the DRMP AVP when relaying request and answer messages. This includes the selection of routes and the ordering of messages relayed.

> Note: The priority information included in the DRMP AVP in request messages applies to both the request message and, by default, answer message associated with the transaction.

While done only in exceptional circumstances, Diameter agents MAY modify priority information when relaying request and answer messages.

> Note: There might be scenarios where a Diameter agent does modify priority information. For instance, an edge agent might need to modify the priority values set by an adjacent operator.

While done only in exceptional circumstances, Diameter agents MAY add priority information when relaying request and answer messages.

> Note: There might be scenarios where a Diameter endpoint does not support the DRMP mechanism and agents insert priority information for that non-supporting endpoint.

Diameter endpoints MAY use routing priority information included in the DRMP AVP when making resource allocation decisions for the transaction associated with the request message that contains the DRMP information.

Diameter endpoints MAY use routing priority information included in the DRMP AVP when making resource allocation decisions for the transaction associated with the answer messages using the DRMP information associated with the transaction.

Diameter endpoints MAY include the DRMP AVP in answer messages. This is done when the priority for the answer message needs to have a different priority than the priority carried in the request message.

When determining the priority to apply to answer messages, Diameter nodes SHOULD use the priority indicated in the DRMP AVP carried in the answer message, if it exists. If there is not DRMP AVP in the answer message then the Diameter node SHOULD use the priority indicated in the DRMP AVP of the associated request message.

> Note: One method to determine what priority to apply to an answer when there is no DRMP AVP in the answer message is to save the priority included in the request message in state associated with the Diameter transaction. Another is to use the Proxy-Info mechanism defined in [RFC6733].

Diameter nodes MUST have a default priority to apply to transactions that do not have an explicit priority set in the DRMP AVP.

In order to guaranty consistent handling of messages from nonupgraded Diameter clients, Diameter nodes SHOULD use the PRIORITY_10 priority as this default priority value.

> PRIORITY_10 is a mid range priority that corresponds to "normal" traffic and thus would be a suitable default for most deployments, while still allowing different Diameter applications to designate other priorities for lower and higher priority traffic.
>
> Note: This does not imply that a DRMP AVP is added to the message. Rather, the message is treated the same as a message that has a DRMP AVP with priority value of PRIORITY_10.

Diameter nodes MUST support the ability for the default priority to be modified through local configuration interfaces.

> Note: There are scenarios where operators might want to specify a different default value for transactions that do not have an explicit priority. In this case, the operator defined local policy would override the use of PRIORITY_10 as the default priority.

When using DRMP priority information, Diameter nodes MUST use the default priority for transactions that do not have priority specified in a DRMP AVP.

Note: This guidance on the handling of messages without a priority does not result in a Diameter agent inserting a DRMP AVP into the message. Rather, it gives guidance on how that specific transaction should be treated when its priority is compared with other requests. When a Diameter agent relays the request it will not insert a DRMP AVP with a priority value of 10.

When setting and using priorities, for all integers x,y in [0,15] treat PRIORITY_<x> as lower priority than PRIOIRTY_<y> when y<x.

Note: As a result PRIORITY_0 is the highest priority.

9. Attribute Value Pairs

This section describes the encoding and semantics of the Diameter Overload Indication Attribute Value Pair (AVP) defined in this document.

9.1. DRMP AVP

The DRMP (AVP code TBD1) is of type Enumerated. The value of the AVP indicates the routing message priority for the transaction. The following values are defined:

PRIORITY_15 15 PRIORITY_15 is the lowest priority.

PRIORITY_14 14 PRIORITY_14 is a higher priority than PRIORITY_15 and a lower priority than PRIORITY_13.

PRIORITY_13 13 PRIORITY_13 is a higher priority than PRIORITY_14 and a lower priority than PRIORITY_12.

PRIORITY_12 12 PRIORITY_12 is a higher priority than PRIORITY_13 and a lower priority than PRIORITY_11.

PRIORITY_11 11 PRIORITY_11 is a higher priority than PRIORITY_12 and a lower priority than PRIORITY_10.

PRIORITY_10 10 PRIORITY_10 is a higher priority than PRIORITY_11 and a lower priority than PRIORITY_9.

PRIORITY_9 9 PRIORITY_9 is a higher priority than PRIORITY_10 and a lower priority than PRIORITY_8.

PRIORITY_8 8 PRIORITY_8 is a higher priority than PRIORITY_9 and a lower priority than PRIORITY_7.

PRIORITY_7 7 PRIORITY_7 is a higher priority than PRIORITY_8 and a lower priority than PRIORITY_6.

PRIORITY_6 6 PRIORITY_6 is a higher priority than PRIORITY_7 and a lower priority than PRIORITY_5.

PRIORITY_5 5 PRIORITY_5 is a higher priority than PRIORITY_6 and a lower priority than PRIORITY_4.

PRIORITY_4 4 PRIORITY_4 is a higher priority than PRIORITY_5 and a lower priority than PRIORITY_3.

PRIORITY_3 3 PRIORITY_3 is a higher priority than PRIORITY_4 and a lower priority than PRIORITY_2.

PRIORITY_2 2 PRIORITY_2 is a higher priority than PRIORITY_3 and a lower priority than PRIORITY_1.

PRIORITY_1 1 PRIORITY_1 is a higher priority than PRIORITY_2 and a lower priority than PRIORITY_0.

PRIORITY_0 0 Priority 0 is the highest priority.

9.2. Attribute Value Pair flag rules

Attribute Name	AVP Code	Section Defined	Value Type	AVP flag rules		
					MUST	MUST NOT
DRMP	TBD1	x.x	Enumerated			V

10. Considerations When Defining Application Priorities

As discussed in Section 1.1, it is important that the definition of priority values used by all applications within a single Diameter administrative domain be done in a consistent and coordinated manner.

The following are some things to be considered when defining the DRMP priorities to be used in Diameter networks which support Diameter nodes handling multiple applications.

1. As with any prioritization scheme, it is possible for higher priority messages to block lower priority messages from ever being handled. In a Diameter network this will often result in those Diameter transactions being retried. This can result in more traffic than the network would have handled without use of the DRMP mechanism.

   One potential guideline to prevent unwanted starving of lower priority messages is to have higher priority messages represent a relatively small portion of messages handled by the Diameter network under normal scenarios.

      Note that there are scenarios, such as first responder messages, where the blocking of lower priority messages is a requirement.

2. When setting priorities for any of the use cases outlined in Section 5 it is important to use the same priority values across applications. For instance, when defining priority for the First Responder use case discussed in Section 5.1 and the Emergency call use case discussed in Section 5.2 use cases, one high priority value might be used for all first responder messages, say PRIORITY_2, and a slightly lower priority value, say PRIORITY_3, might be used for emergency call related messages. These values should be specified for these use cases across all applications used within the Diameter administrative domain.

      Note that the values mentioned here are strictly for illustrative purposes. The actual values used for these use cases are likely to be different.

3. Messages without the DRMP AVP will be given default priority value threatment. This will include messages from Diameter Clients that have not been updated to support the DRMP mechanism. It might also include messages from foreign administrative domains if the DRMP AVPs are stripped from messages crossing the Diameter administrative domains.

4. The process used to introduce the DRMP mechanism into a Diameter network should also be taken into consideration. Messages of the same type within the same application might get different treatment depending on whether those messages are sent from nodes that are upgraded to support the DRMP mechanism versus nodes that have not yet been upgraded to support the DRMP mechanism.

11. IANA Considerations

11.1. AVP codes

The new AVP defined by this specification is listed in Section 9. All AVP codes are allocated from the 'Authentication, Authorization, and Accounting (AAA) Parameters' AVP Codes registry.

11.2. New registries

There are no new IANA registries introduced by this document.

12. Security Considerations

DRMP gives Diameter nodes the ability to influence which requests are throttled during overload scenarios. In addition, DRMP can be used in determining the routing decisions for request messages. Improper use of the DRMP mechanism could result in the malicious Diameter node gaining preferential treatment, by reducing the probability of its requests being throttled, over other Diameter nodes. This would be achieved by the malicious node inserting artificially high priority values.

Diameter does not include features to provide end-to-end authentication, integrity protection, or confidentiality. This opens the possibility that malicious or compromised agents in the path of a request could modify the DRMP AVP to reflect a priority different than that asserted by the sender of the request.

12.1. Potential Threat Modes

The Diameter protocol involves transactions in the form of requests and answers exchanged between clients and servers. These clients and servers may be peers, that is, they may share a direct transport (e.g., Transmission Control Protocol (TCP) or Stream Control Transmission Protocol (SCTP)) connection, or the messages may traverse one or more intermediaries, known as Diameter Agents. Diameter nodes use Transport Layer Security (TLS), Datagram Transport Layer Security (DTLS), or IPsec to authenticate peers, and to provide confidentiality and integrity protection of traffic between peers. Nodes can make authorization decisions based on the peer identities authenticated at the transport layer.

When agents are involved, this presents an effectively transitive trust model. That is, a Diameter client or server can authorize an agent for certain actions, but it must trust that agent to make appropriate authorization decisions about its peers, and so on. Since confidentiality and integrity protection occurs at the

transport layer, agents can read, and perhaps modify, any part of a Diameter message, including the DRMP AVP.

There are several ways an attacker might attempt to exploit the DRMP mechanism. A malicious or compromised Diameter node might insert invalid priority values resulting in either preferential treatment, resulting from higher values, or degraded treatment resulting from lower values, for that node.

A similar attack involves a malicious or compromised Diameter agent changing the priority value resulting in the sending Diameter node getting either preferential or degraded service.

The DRMP mechanism can be used to aid in overload throttling decisions. When this is the case then the above attacks are limited in scope to when one or more Diameter nodes are in an overloaded state.

The DRMP mechanism can also be used to influence the order in which Diameter messages are handled by Diameter nodes. The above attacks have a potentially greater impact in this scenario as the priority indication impacts the handling of all requests at all times, independent of the overload status of Diameter nodes in the Diameter network.

12.2. Denial of Service Attacks

The DRMP mechanism does not open direct denial of service attack vectors. Rather, it introduces a mechanism where a node can gain unwarranted preferential treatment. It also introduces a mechanism where a node can get degraded service in the scenario where a rogue agent changes the priority value included in messages.

12.3. End-to End-Security Issues

The lack of end-to-end integrity features in Diameter [RFC6733] makes it difficult to establish trust in DRMP AVPs received from non-adjacent nodes. Any agents in the message path may insert or modify DRMP AVPs. Nodes must trust that their adjacent peers perform proper checks on overload reports from their peers, and so on, creating a transitive-trust requirement extending for potentially long chains of nodes. Network operators must determine if this transitive trust requirement is acceptable for their deployments. Nodes supporting DRMP MUST give operators the ability to select which peers are trusted to deliver DRMP AVPs, and whether they are trusted to forward the DRMP AVPs from non-adjacent nodes. Diameter nodes MUST strip DRMP AVPs from messages received from peers that are not trusted for DRMP purposes.

It is expected that work on end-to-end Diameter security might make it easier to establish trust in non-adjacent nodes for DRMP purposes. Readers should be reminded, however, that the DRMP mechanism allows Diameter agents to modify AVPs in existing messages that are originated by other nodes. If end-to-end security is enabled, there is a risk that such modification could violate integrity protection. The details of using any future Diameter end-to-end security mechanism with DRMP will require careful consideration, and are beyond the scope of this document.

13. Contributors

The following people contributed substantial ideas, feedback, and discussion to this document:

- Janet P. Gunn

14. References

14.1. 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, <http://www.rfc-editor.org/info/rfc2119>.

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

14.2. Informative References

[RFC7683] Korhonen, J., Ed., Donovan, S., Ed., Campbell, B., and L. Morand, "Diameter Overload Indication Conveyance", RFC 7683, DOI 10.17487/RFC7683, October 2015, <http://www.rfc-editor.org/info/rfc7683>.

[S6a] 3GPP, "Evolved Packet System (EPS); Mobility Management Entity (MME) and Serving GPRS Support Node (SGSN) related interfaces based on Diameter protocol", 3GPP TS 29.272 10.8.0, June 2013.

Author’s Address

Steve Donovan
Oracle
7460 Warren Parkway
Frisco, Texas 75034
United States

Email: srdonovan@usdonovans.com

DIME
Internet-Draft
Intended status: Informational
Expires: December 10, 2016

H. Tschofenig
ARM Limited
J. Korhonen, Ed.
Broadcom Limited
G. Zorn
Network Zen
K. Pillay
Internet Solutions

June 8, 2016

AVP Level Security for Non-neighboring Diameter Nodes: Scenarios and Requirements
draft-ietf-dime-e2e-sec-req-05.txt

Abstract

This specification specifies 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 December 10, 2016.

Copyright Notice

Table of Contents

1. Introduction

The Diameter base protocol specification [2] defines security protection between neighboring Diameter peers. The Diameter mandates that peer connections must be protected by TLS (for TCP) [6], DTLS (for SCTP) [7] or using security mechanisms that are independent of Diameter such as IPsec [5]. 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. It should be understood that TLS/DTLS/IPsec in Diameter context does not provide end-to-end security unless the Diameter nodes are direct peers i.e., neighboring Diameter nodes. The current Diameter security is realized hop-by-hop.

The need to also offer additional security protection of Attribute Value Pairs (AVP) between non-neighboring Diameter nodes was recognized very early in the work on Diameter. This led 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) and new security building blocks were developed.

This document specifies requirements for developing a solution to protect Diameter AVPs between non-neighboring Diameter nodes.

2. Terminology

The key words 'MUST', 'MUST NOT', 'REQUIRED', 'SHALL', 'SHALL NOT', 'SHOULD', 'SHOULD NOT', 'RECOMMENDED', 'MAY', and 'OPTIONAL' in this documents are to be interpreted as described in RFC 2119 [1].

This document re-uses terminology from the Diameter base specification [2].

In the figures below Attribute Value Pair (AVP) refers to an unprotected AVP and {AVP}k refers to an AVP that experiences security protection (using key "k") without further distinguishing between integrity and confidentiality protection.

The following terms are also used in this document:

AAA Broker

An entity that manages AAA traffic between roaming partner networks.

AAA Broker Network

A network operated by an AAA Broker, which consists of necessary AAA functions to provide AAA brokering services for its customer AAA networks.

Diameter Firewall

A Diameter firewall is a proxy (or a relay) agent that acts similarly to conventional IP traffic firewalls but only at the Diameter AVP and command level. A Diameter firewall may, for example, discard security policy offending AVPs from traversing through it. The Diameter firewall may even discard entire Diameter messages based on the security policy.

3. Security Threats

The following description aims to illustrate various security threats that raise the need for protecting Diameter Attribute-Value Pairs (AVPs). Figure 1 illustrates an example of Diameter based roaming

architecture in which Diameter clients within the visited networks need to interact with Diameter servers in the home domain. AAA domains are interconnected using a Diameter-based AAA interconnection network labeled as AAA Broker.

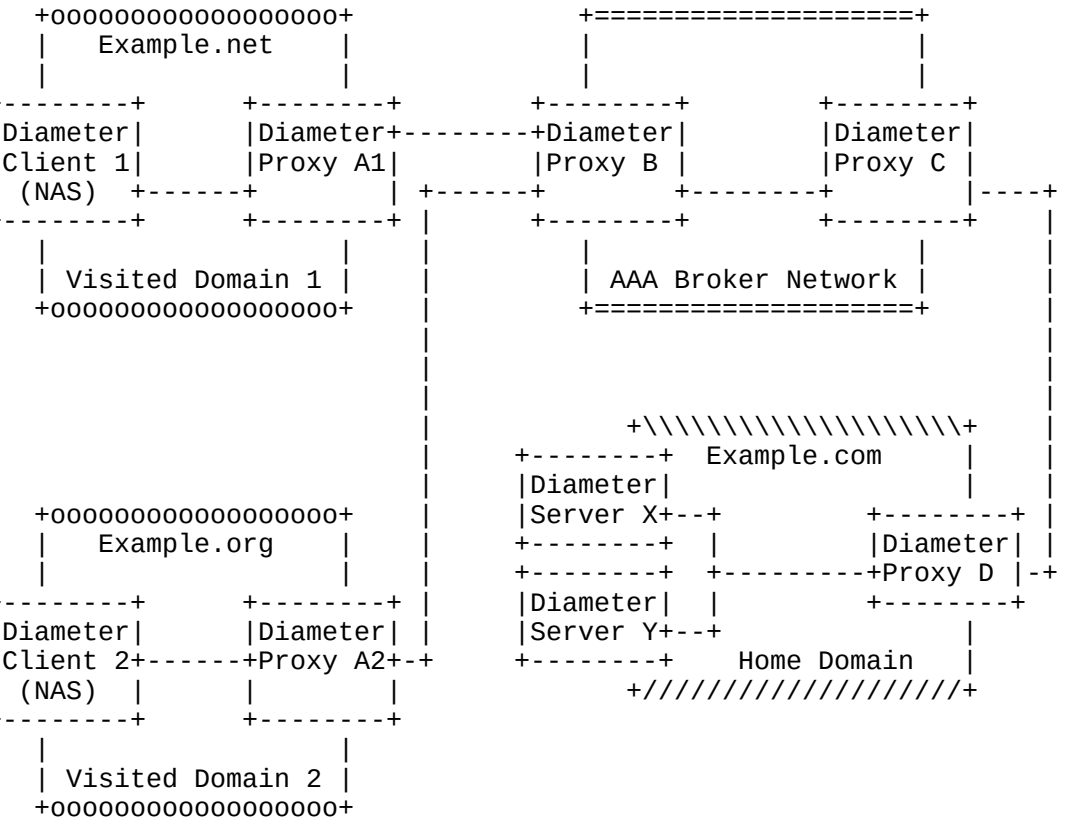

Figure 1: Example Diameter Deployment.

Eavesdropping: Some Diameter applications carry information that is only intended for consumption by end points, either by the Diameter client or by the Diameter server but not by intermediaries. As an example, consider the Diameter EAP application [4] that allows the transport of keying material between the Diameter server to the Diameter client (using the EAP-Master-Session-Key AVP) for the protection of the air interface (i.e., the wireless link) between the end device (such as a mobile phone; not shown in the figure) and the Network Access Server (NAS). The content of the EAP-Master-Session-Key AVP should

benefit from protection against eavesdropping by intermediaries. Other AVPs, for example those listed in Section 13.3 of [2], might also carry sensitive personal data that, when collected by intermediaries, allow for traffic analysis.

In context of the deployment shown in Figure 1 the adversary could, for example, be in the AAA broker network.

Injection and Manipulation: The Diameter base protocol specification mandates security protection between neighboring nodes but Diameter agents may be compromised or misconfigured and inject or manipulate AVPs. To detect such actions additional security protection needs to be applied at the Diameter layer.

Nodes that could launch such an attack are any Diameter agents along the end-to-end communication path.

Impersonation: Imagine a case where a Diameter message from Example.net contains information claiming to be from Example.org. This would either require strict verification at the edge of the AAA broker network or cryptographic assurance at the Diameter layer to prevent a successful impersonation attack.

Any Diameter realm could launch such an attack aiming for financial benefits or to disrupt service availability.

4. Scenarios for Diameter AVP-Level Protection

This scenario outlines a number of cases for deploying security protection of individual Diameter AVPs.

In the first scenario, shown in Figure 2, end-to-end security protection is provided between the Diameter client and the Diameter server with any number of intermediate Diameter agents. Diameter AVPs exchanged between these two Diameter nodes may be protected end-to-end (notation '{AVP}k') or unprotected (notation 'AVP').

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. Typically, the proxy enforcing the security protection belongs to the same domain as the Diameter client/server without end-to-end security features.

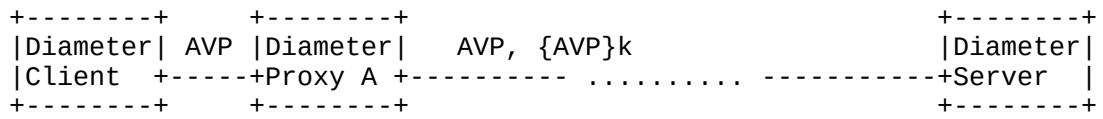

Figure 3: Middle-to-End Diameter AVP Security Protection.

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 fourth and the 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, except for the security enabled proxies obviously. Second, the key management is also simplified since fewer number of keys need to be negotiated and provisioned. The assumption here is that the number of security enabled proxies would be significantly less than unprotected Diameter nodes in the installed base.

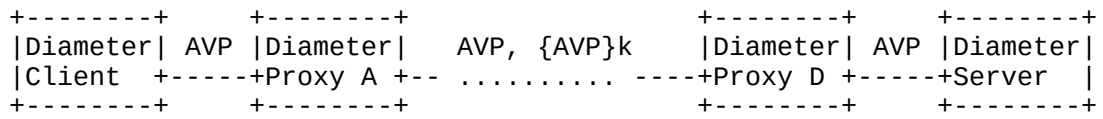

Figure 5: Middle-to-Middle Diameter AVP Security Protection.

5. Requirements

Requirement #1: The solution MUST support an extensible set of cryptographic algorithms.

Motivation: Solutions MUST be able to evolve 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: The solution MUST support confidentiality, integrity, and data-origin authentication. Solutions for integrity protection MUST work in a backwards-compatible way with existing Diameter applications and therefore be able to traverse legacy proxy and relay agents.

Requirement #3: The solution MUST support replay protection.

Requirement #4: The solution MUST support the ability to delegate security functionality to another entity

Motivation: As described in Section 4 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: The solution 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 must be possible to apply security protection selectively. Furthermore, there are AVPs that must not be confidentiality protected but may still be integrity protected such as those required for Diameter message routing.

Requirement #6: The solution MUST define 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: The solution 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 MUST be included in the overall solution framework.

However, 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.

6. Security Considerations

This entire document focused on the discussion of new functionality for securing Diameter AVPs selectively between non-neighboring nodes.

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 the most benefits for existing deployments with minimal overhead and (potentially) operating in a full-backwards compatible manner.

7. IANA Considerations

This document does not require actions by IANA.

8. Acknowledgments

We would like to thank Guenther Horn, Martin Dolly, Steve Donovan, Lionel Morand and Tom Taylor (rest in peace Tom) for their review comments.

The authors also thank Qin Wu, Christer Holmberg, Ben Campbell and Radia Perlman who provided additional reviews during the Last Call.

9. References

9.1. Normative References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

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

9.2. 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.

[4] Eronen, P., Ed., Hiller, T., and G. Zorn, "Diameter Extensible Authentication Protocol (EAP) Application", RFC 4072, DOI 10.17487/RFC4072, August 2005, <http://www.rfc-editor.org/info/rfc4072>.

[5] Kent, S. and K. Seo, "Security Architecture for the Internet Protocol", RFC 4301, DOI 10.17487/RFC4301, December 2005, <http://www.rfc-editor.org/info/rfc4301>.

[6] Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, DOI 10.17487/RFC5246, August 2008, <http://www.rfc-editor.org/info/rfc5246>.

[7] Tuexen, M., Seggelmann, R., and E. Rescorla, "Datagram Transport Layer Security (DTLS) for Stream Control Transmission Protocol (SCTP)", RFC 6083, DOI 10.17487/RFC6083, January 2011, <http://www.rfc-editor.org/info/rfc6083>.

Authors' Addresses

Hannes Tschofenig
ARM Limited
Austria

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

Jouni Korhonen (editor)
Broadcom Limited
3151 Zanker Rd.
San Jose, CA 95134
USA

Email: jouni.nospam@gmail.com

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

Email: glenzorn@gmail.com

Kervin Pillay
Internet Solutions
South Africa

Email: kervin.pillay@gmail.com

Diameter Maintenance and Extensions (DIME)
Internet-Draft
Intended status: Standards Track
Expires: September 22, 2016

M. Jones

M. Liebsch

L. Morand

March 21, 2016

Diameter Group Signaling
draft-ietf-dime-group-signaling-06.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.

Table of Contents

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", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

This document uses terminology defined [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 mid-session. For example, if a user's subscription profile changes mid-session, a Diameter server may remove the session from its current group and assign the session to a different group that is more appropriate for the new subscription profile.

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

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. The server issues a single Session Termination Request (STR) command , identifying the group of sessions which are to be terminated. The Diameter client treats the STR as group command and initiates termination of all sessions associated with the identified group. Subsequently, the client confirms 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 Diameter node decides to create a new session group, e.g. to group all sessions which share certain characteristics, the 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 constrain the permission to add or remove a session to/from a session group to the group owner, instead each node can add a session to any known group or remove a session from a group. A session group is deleted and its identifier released after the last session has been removed from the session group. Also the modification of groups in terms of moving a session from one session group to a different session group is permitted to any Diameter node. A Diameter node 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. Prerequisite for deletion of a session group is that the Diameter node created the session beforehand, hence the node became the group owner.

The enforcement of more constrained permissions is left to the specification of a particular group signaling enabled Diameter application and compliant implementations of such application must enforce the associated permission model. Details about enforcing a more constraint permission model are out of scope of this specification. 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.

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 a different Diameter node created 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

*) Editors’ note: The protocol specification in this document does not consider overruling a node’s assignment of a session to a session group. Here, overruling is to be understood as a node changing the group(s) assignment as per the node’s request. Group signaling enabled applications may take such protocol support and associated protocol semantics into account in their specification. One exception is adopted in this specification, which allows a Diameter server to reject a group assignment as per the client’s request.

4. Protocol Description

4.1. Session Grouping Capability Discovery

Diameter nodes should assign a session to a session group and perform session group operations with a node only after having ensured that the node announced associated support beforehand.

4.1.1. Explicit Capability Discovery

New Diameter applications may consider support for Diameter session grouping and for performing group commands during the standardization process. Such applications provide intrinsic discovery for the support of group commands and announce this capability through the assigned application ID.

System- and deployment-specific means, as well as out-of-band mechanisms for capability exchange 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. Implicit Capability Discovery

If no explicit mechanism for capability discovery is deployed to enable Diameter nodes to learn about nodes' capability to support session grouping and group commands, a Diameter node SHOULD append the Session-Group-Capability-Vector AVP to any Diameter messages exchanged with its nodes to announce its capability to support session grouping and session group operations. Implementations following the specification as per this document 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 Diameter node maintains a log to remember the node's capability to support group commands.

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 application to determine the policy of session grouping. In case 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. 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 should be 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 [RFC4005], containing one or more session group identifiers. Each of these groups needs to 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

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 includes in the request the Session-Group-Info AVP in the request including the Session-Group-Control-Vector AVP with 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 comprises at least one Session-Group-Info AVP having the SESSION_GROUP_ALLOCATION_ACTION flag set 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 but be treated as single session.

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. In case one or multiple identified session groups are not already stored by the server, the server must store the new 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 [RFC4005], 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 adds 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 [RFC4005], 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.

If the Diameter server receives a command request from a Diameter client and the command comprises one or multiple Session-Group-Info AVPs and none of them including 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 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 comprises 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 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 and continue to treat 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.

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 comprises 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. In case 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 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. The client subsequently sends 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 send at least on 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 being 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 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 follow up message exchanges should be performed by appending a single instance of the Group-Response-Action AVP. Even if the request includes multiple instances of a Session-Group-Info AVP, the request MUST NOT comprise more than a single instance of a Group-Response-Action AVP. If the sender wants the receiver to perform follow up exchanges with a single command for all impacted groups, the sender sets the value of the Group-Response-Action AVP to ALL_GROUPS (1). If follow up message exchanges should be performed on a per-group basis in case multiple groups are identified in the group command, the value of the Group-Response-Action AVP is set to PER_GROUP (2). A value set to PER_SESSION (3) indicates to the receiver that all follow up exchanges should be performed using a single message for each impacted session.

If the sender sends a request including the Group-Response-Action AVP set to ALL_GROUPS (1) or PER_GROUP (2), it MUST expect some delays 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 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.

The Diameter node receiving a request which requests performing the command to at least on session group SHOULD perform follow up message exchanges according to the value identified in the Session-Group-Info AVP.

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 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 Session 7.5 of [RFC6733].

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. 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 Proxies Agents

In case of a present stateful Proxy Agent between a Diameter client and a Diameter server, this specification assumes that the Proxy Agent is aware of session groups and session group handling. 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 case a 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.

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 capabilities 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 comprise 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 and eternally unique, as it is meant to uniquely identify a group of Diameter sessions without reference to any other information.

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 application:

ALL_GROUPS (1)
: Follow up exchanges should be performed with a single message exchange for all impacted groups.

PER_GROUP (2)
: Follow up exchanges should be performed with a message exchange for each impacted group.

PER_SESSION (3)
: Follow up exchanges should be performed with a 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].

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

The AVPs are defined in Section 7.

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. The DIME working group is currently working on solutions for providing end-to-end security features. When available, these features should 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 assume 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 for the thorough review and comments on the adopted 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, <http://www.rfc-editor.org/info/rfc2119>.

[RFC4005] Calhoun, P., Zorn, G., Spence, D., and D. Mitton, "Diameter Network Access Server Application", RFC 4005, DOI 10.17487/RFC4005, August 2005, <http://www.rfc-editor.org/info/rfc4005>.

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

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	Send GASA with Result-Code = SUCCESS, Send STR	Discon

	client will comply with request to end the session	per session	
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 Group-Response-Action = PER_SESSION, session is assigned to received group(s) and client will perform subsequent re-auth	Send GRAA, Send service specific re-auth req per session	Pending
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	Discon. user/device	Idle

received.

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

Internet Engineering Task Force
Internet-Draft
Intended status: Standards Track
Expires: September 19, 2016

B. Campbell
S. Donovan, Ed.
Oracle
JJ. Trottin
Nokia

March 18, 2016

Diameter Load Information Conveyance
draft-ietf-dime-load-02

Abstract

This document defines a mechanism for sharing of Diameter load information. [RFC7068] describes requirements for Overload Control in Diameter. This includes a requirement to allow Diameter nodes to send "load" information, even when the node is not overloaded. The Diameter Overload Information Conveyance (DOIC) [RFC7683] solution describes a mechanism meeting most of the requirements, but does not currently include the ability to send load information.

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 19, 2016.

Copyright Notice

Table of Contents

1. Introduction

[RFC7068] describes requirements for Overload Control in Diameter [RFC6733]. The DIME working group has finished the Diameter Overload Information Conveyance (DOIC) mechanism [RFC7683]. As currently specified, DOIC fulfills some, but not all, of the requirements.

In particular, DOIC does not fulfill Req 24, which requires a mechanism where Diameter nodes can indicate their current load, even if they are not currently overloaded. DOIC also does not fulfill Req 23, which requires that nodes that divert traffic away from overloaded nodes be provided with sufficient information to select targets that are most likely to have sufficient capacity.

There are several other requirements in [RFC7068] that mention both overload and load information that are only partially fulfilled by DOIC.

The DIME working group explicitly chose not to fulfill these requirements in DOIC due to several reasons. A principal reason was that the working group did not agree on a general approach for conveying load information. It chose to progress the rest of DOIC, and deferred load information conveyance to a DOIC extension or a separate mechanism.

This document defines a mechanism that addresses the load-related requirements from RFC 7068.

2. Terminology and Abbreviations

DOIC

Diameter Overload Information Conveyance ([RFC7683])

Load

The relative capacity of a Diameter node. A low load level indicates that the Diameter node is under utilized. A high load level indicates that the node is closer to being fully utilized.

Offered Load

The actual traffic sent to the reporting node after overload abatement and routing decisions are made.

Reporting, Reacting Node

Reporting node and reacting node terminology is defined in [RFC7683].

Routing Information

Routing Information - Routing information referred to in this document can include the Routing and Peer tables defined in RFC 6733. It can also include other implementation specific tables used to store load information. This document does not define the structure of such tables.

3. Conventions Used in This Document

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

RFC 2119 [RFC2119] interpretation does not apply for the above listed words when they are not used in all-caps format.

4. Background

4.1. Differences between Load and Overload information

Previous discussions of how to solve the load-related requirements in [RFC7068] have shown that people did not had an agreed-upon concept of how "load" information differs from "overload" information. While the two concepts are highly interrelated, in the opinion of the authors, there are two primary differences. First, a Diameter node always has a load. At any given time that load maybe effectively zero, effectively fully loaded, or somewhere in between. In contrast, overload is an exceptional condition. A node only has overload information when it is in an overloaded state. Furthermore, the relationship between a node’s load level and overload state at any given time may be vague. For example, a node may normally operate at a "fully loaded" level, but still not be considered overloaded. Another node may declare itself to be "overloaded" even though it might not be fully "loaded".

Second, Overload information, in the form of a DOIC Overload Report (OLR) [RFC7683] indicates an explicit request for action on the part of the reacting node. That is, the OLR requests that the reacting node reduce the offered load -- the actual traffic sent to the reporting node after overload abatement and routing decisions are made -- by an indicated amount or to an indicated level. Effectively, DOIC provides a contract between the reporting node and the reacting node.

In contrast, load is informational. That is, load information can be considered a hint to the recipient node. That node may use the load information for load balancing purposes, as an input to certain overload abatement techniques, to make inferences about the likelihood that the sending node becomes overloaded in the immediate future, or for other purposes.

None of this prevents a Diameter node from deciding to reduce the offered load based on load information. The fundamental difference is that an overload report requires that reduction. It is also reasonable for a Diameter node to decide to increase the offered load based on load information.

4.2. How is Load Information Used?

[RFC7068] contemplates two primary uses for load information. Req 23 discusses how load information might be used when performing diversion as an overload abatement technique, as described in [RFC7683]. When a reacting node diverts traffic away from an overloaded node, it needs load information for the other candidates for that traffic in order to effectively load balance the diverted load between potential candidates. Otherwise, diversion has a greater potential to drive other nodes into overload.

Req 24 discusses how Diameter load information might be used when no overload condition currently exists. Diameter nodes can use the load information to make decisions to try to avoid overload conditions in the first place. Normal load-balancing falls into this category. A node might also take other proactive steps to reduce offered load based on load information, so that the loaded node never goes into overload in the first place.

If the loaded nodes are Diameter servers (or clients in the case of server-to-client transactions), both of these uses are most effectively accomplished by a Diameter node that performs server selection. Typically, server selection is performed by a node (a client or an agent) that is an immediate peer of the server. However, there are scenarios (see Appendix A) where a client or proxy that is not the immediate peer to the selected servers performs server selection. In this case, the client or proxy enforces the server selection by inserting a Destination-Host AVP.

> For example, a Diameter node (e.g. client) can use a redirect agent to get candidate destination host addresses. The redirect agent might return several destination host addresses, from which the Diameter node selects one. The Diameter node can use load information received from these hosts to make the selection.

Just as load information can be used as part of server selection, it can also be used as input to the selection of the next-hop peer to which a request is to be routed.

It should be noted that a Diameter node will need to process both Load reports and Overload reports from the same Diameter node. The reacting node for the Overload report always has the responsibility to reduce the amount of Diameter traffic sent to the overloaded node. If, or how, the reacting node uses Load information to achieve this is left as an implementation decision.

5. Solution Overview

The mechanism defined here for the conveyance of load information is similar in some ways to the mechanism defined for DOIC and is different in other ways.

As with DOIC, load information is conveyed by piggy-backing the load AVPs on existing Diameter applications.

There are two primary differences. First, there is no capability negotiation process for load. The sender of the load information is sending it with the expectation that any supporting nodes will use it when making routing decisions. If there are no nodes that support the Load mechanism then the load information is ignored.

The second big difference between DOIC and Load is visibility of the DOIC or Load information within a Diameter network. DOIC information is sent end-to-end resulting in the ability of all nodes in the path of the answer message that carries the OC-OLR AVP to act on the information. The DOIC overload reports much remain in the message all the way from the reporting node to the node that is the target for the answer message.

For the Load mechanism there are two types of load reports.

The first is the load of the endpoint sending the answer message. This load report is carried end-to-end to enable any nodes that make server selection decisions to use the load status of the sending endpoint as part of the server selection decision.

The second type of load report is a peer report. This report is used by Diameter nodes as part of the logic to select the next hop Diameter node and, as such, do not have significance beyond the peer node. These load reports are removed by the first supporting Diameter node to receive the report.

Because load reports can traverse Diameter nodes that do not support the Load mechanism, it is necessary to include the identity of the node to which the load report applies as part of the load report. This allows for a Diameter node to verify that a load report applies to its peer or if it should be ignored.

The load report includes the relative load of the sending node. This relative load is specified in a manner consistent with that defined for DNS SRV [RFC2782].

The goal is make it possible to use both the load values received as a part of the Diameter Load mechanism and weight values received as a result of a DNS SRV query. As a result, the Diameter load value has a range of 0-65535. This value and DNS SRV weight values are then used in a distribution algorithm similar to that specified in [RFC2782].

The DNS SRV distribution algorithm results in more messages being sent to a node with a higher weight value. As a result, a higher Diameter load value indicates a LOWER load on the sending node. A node that is heavily loaded sends a lower Diameter load value. Stated another way, a node that has zero load would have a load value of 65535. A node that is 100% loaded would have a load value of 0.

The distribution algorithm used by Diameter nodes supporting the Diameter Load mechanism is an implementation decision but it needs to result in similar behavior as the algorithm specified in [RFC2782].

The method for calculating the load value included in the load report is also left as an implementation decision.

The frequency for sending of load reports is also left as an implementation decision. The sending node might choose to send load reports in all messages or it might choose to only send load reports when the load value has changed by some implementation specific amount. The important consideration is that all nodes needing the load information have a sufficiently accurate view of the nodes load.

5.1. Theory of Operation

This section outlines how the Diameter Load mechanism is expected to work.

For this discussion, assume the following Diameter network configuration:

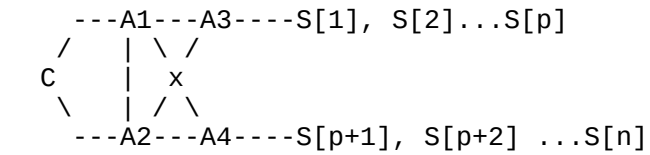

Figure 1: Example Diameter Network

Note that in this diagram, S[1], S[2] through S[p] are peers to A3. S[p+1], S[p+2] through S[n] are peers to A4.

Also assume that the request for a Diameter transaction takes the following path:

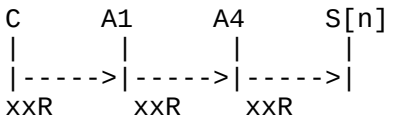

Figure 2: Request Message Path

When sending the answer message, an endpoint node that supports the Diameter Load mechanism includes it’s own load information in the answer message. Because it is a Diameter endpoint it includes a HOST load report.

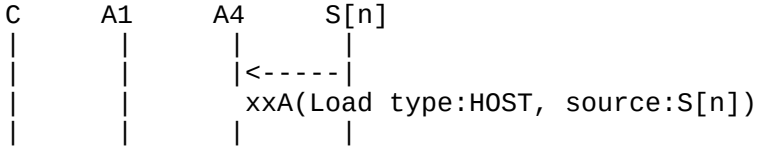

Figure 3: Answer Message from S[n]

If Agent A4 supports the Load mechanism then it will verify that the load information received is valid. For a HOST load report this is achieved by matching the identity included in the load information with the identity of the host node from which the answer message was received.

Note: If A4 does not support the Load mechanism then it will relay the answer message without doing any processing on the load information. In this case the load information AVPs will be relayed without change.

If the identity included in the load information AVPs matches the identity of the host from which the load information is received then Agent A4 stores the load information for S[n] in its routing information.

Because the load report is an HOST load report, A4 leaves the load report in the message it relays.

A4 then calculates its own load information and inserts load information AVPs of type PEER in the message before sending the message to A1:

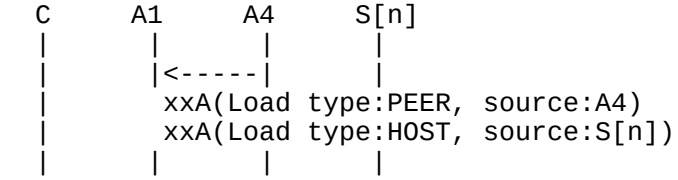

Figure 4: Answer Message from A4

If A1 supports the Load mechanism then it processes each of the Load reports it receives separately.

For the PEER load report, A1 first determines if the source of the report indicated in the load report matches the DiameterIdentity of the Diameter node from which the request was received. If the identities do not match then the PEER load report is discarded. If the identities match then A1 saves the load information in its routing information for routing of subsequent request messages. In both cases A1 strips the PEER load report from the message.

For the HOST load report, A1's actions depend on whether A1 is responsible for doing server selection. If A1 is not doing server selection then A1 ignores the HOST load report. If A1 is responsible for doing server selection then it stores the load information for S[n] in its routing information for the handling of subsequent request messages. In both cases A1 leaves the HOST report in the message.

A1 then calculates its own load information and inserts load information AVPs of type PEER in the message before sending the message to A1:

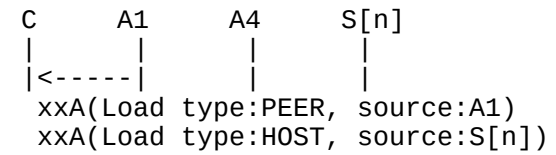

Figure 5: Answer Message from A1

As with A1, C processes each load report separately.

For the PEER load report, C follows the same procedure as A1 for determining if the Load report was received from the peer from which the report was sent and, when finding it does, stores the load information for use when making future routing decisions.

For the HOST load report, C saves the load information only if it is responsible for doing server selection.

The Load information received by all nodes is then used for routing of subsequent request messages.

6. Load Mechanism Procedures

This section defines the normative behaviors for the Load mechanism.

6.1. Reporting Node Behavior

This section defines the procedures of Diameter reporting nodes that generate load reports.

6.1.1. Endpoint Reporting Node Behavior

A Diameter endpoint that supports the Diameter Load mechanism MUST include a load report of type HOST in sufficient answer messages to ensure that all consumers of the load information receive timely updates.

The Diameter endpoint MUST include it’s own DiameterIdentity in the Source-ID AVP included in the Load AVP.

The Diameter endpoint MUST include a Load-Type AVP of type HOST in the Load AVP.

The Diameter endpoint MUST include its load value in the Value AVP in the load AVP.

The method for determining the load value included in the load report is an implementation decision.

The frequency of sending load reports is an implementation decision.

For instance, if the only consumer of the load reports is the endpoints peer then the endpoint can choose to only include a load report when the load of the endpoint has changed by a meaningful percentage. If there are consumers of the endpoint load report other then the endpoints peer (this will be the case if other nodes are responsible for server selection) then the endpoint might choose to include load reports in all answer messages as a way of ensuring that all nodes doing server selection get accurate load information.

6.1.2. Agent Reporting Node Behavior

A Diameter agent that supports the Diameter Load mechanism MUST include a PEER load report in sufficient answer messages to ensure that all users of the load information receive timely updates.

The Diameter agent MUST include it’s own DiameterIdentity in the Source-ID AVP included in the Load AVP.

The Diameter agent MUST include a Load-Type AVP of type PEER in the Load AVP.

The Diameter agent MUST include its load value in the Value AVP in the load AVP.

The method for determining the load value included in the load report is an implementation decision.

The frequency of sending load reports is an implementation decision.

Note: In the case of peer load reports it is only necessary to include load reports when the load value has changed by some meaningful value, as long as the agent insures that all peers receive the report. It is also acceptable to include the load report in every answer message handled by the Diameter agent.

6.2. Receiving Node Behavior

This section defines the behavior of Diameter nodes processing load reports.

A Diameter node MUST be prepared to process load reports of type HOST and of type PEER, as indicated in the Load-Type AVP included in the

Load AVP received in the same answer message or from multiple answer messages.

Note that the node needs to be able to handle messages with no load reports, messages with just a PEER load report, messages with just an HOST load report and messages with both types of load reports.

If the Diameter node is not responsible for doing server selection then it SHOULD ignore load reports of type HOST.

If the Diameter node is responsible for doing server selection then it SHOULD save the load value included in the Value AVP included in the Load AVP of type HOST in its routing information.

If the Diameter node receives a Load report of type PEER then the Diameter node MUST determine if the Load report was inserted into the answer message by the peer from which the message was received. This is achieved by comparing the DiameterIdentity associated with the connection from which the message was received with the DiameterIdentity included in the Source-ID AVP in the Load report.

If the Diameter node determines that the Load report of type PEER was not received from the peer that sent or relayed the answer message then the node MUST ignore the Load report.

If the Diameter node determines that the Load report of type PEER was received from the peer that sent or relayed the answer message then the node SHOULD save the load information in its routing information.

How a Diameter node uses load information for making routing decisions is an implementation decision. However, the distribution algorithm MUST result in similar behavior as the algorithm described for the use of weigth values in [RFC2782].

6.3. Extensibility

The Load mechanism can be extended to include additional information in the load reports.

Any extension may define new AVPs for use in Load reports. These new AVPs SHOULD be defined to be extensions to the Load AVPs defined in this document.

[RFC6733] defined Grouped AVP extension mechanisms apply. This allows, for example, defining a new feature that is mandatory to be understood even when piggybacked on an existing application.

As with any Diameter specification, [RFC6733] requires all new AVPs to be registered with IANA. See Section 9 for the required procedures.

7. Attribute Value Pairs

The section defines the AVPs required for the Load mechanism.

7.1. Load AVP

The Load AVP (AVP code TBD1) is of type Grouped and is used to convey load information between Diameter nodes.

```
 Load ::= < AVP Header: TBD1 >
          [ Load-Type ]
          [ Load-Value ]
          [ SourceID ]
        * [ AVP ]
```

7.2. Load-Type AVP

The Load-Type AVP (AVP code TBD2) is of type Enumerated. It is used to convey the type of Diameter node that sent the load information. The following values are defined:

HOST 0 The load report is for a host.

PEER 1 The load report is for a peer.

7.3. Load-Value AVP

The Load-Value AVP (AVP code TBD3) is of type Unsigned64. It is used to convey relative load information about the sender of the load report.

The Load-Value AVP is specified in a manner similar to the weight value in DNS SRV ([RFC2782]).

The Load-Value has a range of 0-65535.

A higher value indicates a lower load on the sending node. A lower value indicates that the sending node is heavily loaded.

> Stated another way, a node that has zero load would have a load value of 65535. A node that is 100% loaded would have a load value of 0.

7.4. SourceID AVP

The SourceID AVP is defined in [I-D.ietf-dime-agent-overload]. It is used to identify the Diameter node that sent the Load report.

7.5. Attribute Value Pair flag rules

Attribute Name	AVP Code	Section Defined	Value Type	AVP flag rules: MUST	AVP flag rules: MUST NOT
Load	TBD1	x.1	Grouped		V
Load-Type	TBD2	x.2	Enumerated		V
Load-Value	TBD3	x.3	Unsigned64		V
SourceID	TBD4	x.4	DiameterIdentity		V

As described in the Diameter base protocol [RFC6733], the M-bit usage for a given AVP in a given command may be defined by the application.

8. Security Considerations

Load information may be sensitive information in some cases. Depending on the mechanism, an unauthorized recipient might be able to infer the topology of a Diameter network from load information. Load information might be useful in identifying targets for Denial of Service (DoS) attacks, where a node known to be already heavily loaded might be a tempting target. Load information might also be useful as feedback about the success of an ongoing DoS attack.

Any load information conveyance mechanism will need to allow operators to avoid sending load information to nodes that are not authorized to receive it. Since Diameter currently only offers authentication of nodes at the transport level, any solution that sends load information to non-peer nodes might require a transitive-trust model.

9. IANA Considerations

9.1. AVP Codes

New AVPs defined by this specification are listed in Section Section 7. All AVP codes are allocated from the 'Authentication, Authorization, and Accounting (AAA) Parameters' AVP Codes registry.

9.2. New Registries

This document makes no new registry requests of IANA.

10. References

10.1. Normative References

[I-D.ietf-dime-agent-overload]
Donovan, S., "Diameter Agent Overload", draft-ietf-dime-agent-overload-02 (work in progress), August 2015.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <http://www.rfc-editor.org/info/rfc2119>.

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

[RFC7068] McMurry, E. and B. Campbell, "Diameter Overload Control Requirements", RFC 7068, DOI 10.17487/RFC7068, November 2013, <http://www.rfc-editor.org/info/rfc7068>.

[RFC7683] Korhonen, J., Ed., Donovan, S., Ed., Campbell, B., and L. Morand, "Diameter Overload Indication Conveyance", RFC 7683, DOI 10.17487/RFC7683, October 2015, <http://www.rfc-editor.org/info/rfc7683>.

10.2. Informative References

[RFC2782] Gulbrandsen, A., Vixie, P., and L. Esibov, "A DNS RR for specifying the location of services (DNS SRV)", RFC 2782, DOI 10.17487/RFC2782, February 2000, <http://www.rfc-editor.org/info/rfc2782>.

Appendix A. Topology Scenarios

This section presents a number of Diameter topology scenarios, and discusses how load information might be used in each scenario. Nothing in this section should be construed to mean that a given scenario is in scope for this effort, or even a good idea. Some scenarios might be considered as not relevant in practice and subsequently discarded.

A.1. No Agent

Figure 6 shows a simple client-server scenario, where a client picks from a set of candidate servers available for a particular realm and application. The client selects the server for a given transaction using the load information received from each server.

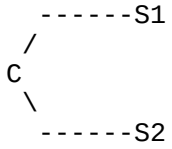

Figure 6: Basic Client Server Scenario

If a node supports dynamic discovery, it will not obtain load information from the nodes with which it has no Diameter connection established. Nevertheless it might take into account the load information from the other nodes to decide to add connections to new nodes with the dynamic discovery mechanism.

Note: The use of dynamic connections needs to be considered.

A.2. Single Agent

Figure 7 shows a client that sends requests to an agent. The agent selects the request destination from a set of candidate servers, using load information received from each server. The client does not need to receive load information, since it does not select between multiple agents.

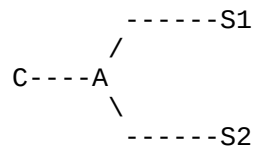

Figure 7: Simple Agent Scenario

A.3. Multiple Agents

Figure 8 shows a client selecting between multiple agents, and each agent selecting from multiple servers. The client selects an agent based on the load information received from each agent. Each agent selects a server based on the load information received from its servers.

This scenario adds a complication that one set of servers may be more loaded than the other set. If, for example, S4 was the least loaded server, C would need to know to select agent A2 to reach S4. This might require C to receive load information from the servers as well as the agents. Alternatively, each agent might use the load of its servers as an input into calculating its own load, in effect aggregating upstream load.

Similarly, if C sends a host-routed request [RFC7683], it needs to know which agent can deliver requests to the selected server. Without some special, potentially proprietary, knowledge of the topology upstream of A1 and A2, C would select the agent based on the normal peer selection procedures for the realm and application, and perhaps consider the load information from A1 and A2. If C sends a request to A1 that contains a Destination-Host AVP with a value of S4, A1 will not be able to deliver the request.

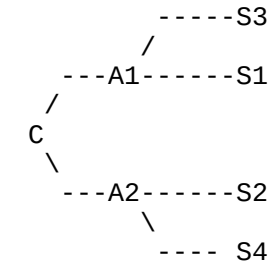

Figure 8: Multiple Agents and Servers

A.4. Linked Agents

Figure 9 shows a scenario similar to that of Figure 8, except that the agents are linked, so that A1 can forward a request to A2, and vice-versa. Each agent could receive load information from the linked agent, as well as its connected servers.

This somewhat simplifies the complication from Figure 8, due to the fact that C does not necessarily need to choose a particular agent to reach a particular server. But it creates a similar question of how, for example, A1 might know that S4 was less loaded than S1 or S3. Additionally, it creates the opportunity for sub-optimal request paths. For example [C,A1,A2,S4] vs. [C,A2,S4].

A likely application for linked agents is when each agent prefers to route only to directly connected servers and only forwards requests to another agent under exceptional circumstances. For example, A1 might not forward requests to A2 unless both S1 and S3 are overloaded. In this case, A1 might use the load information from S1 and S3 to select between those, and only consider the load information from A2 (and other connected agents) if it needs to divert requests to different agents.

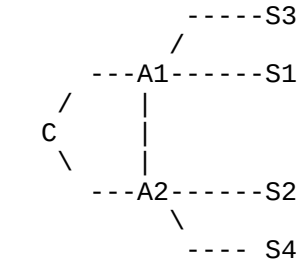

Figure 9: Linked Agents

Figure 10 is a variant of Figure 9. In this case, C1 sends all traffic through A1 and C2 sends all traffic through A2. By default, A1 will load balance traffic between S1 and S3 and A2 will load balance traffic between S2 and S4.

Now, if S1 S3 are significantly more loaded than S2 S4, A1 may route some C1 traffic to A2. This is non optimal path but allows a better load balancing between the servers. To achieve this, A1 needs to receive some load info from A2 about S2/S4 load.

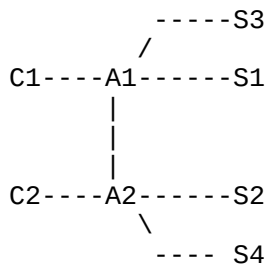

Figure 10: Linked Agents

A.5. Shared Server Pools

Figure 11 is similar to Figure 9, except that instead of a link between agents, each agent is linked to all servers. (The links to each set of servers should be interpreted as a link to each server. The links are not shown separately due to the limitations of ASCII art.)

In this scenario, each agent can select among all of the servers, based on the load information from the servers. The client need only be concerned with the load information of the agents.

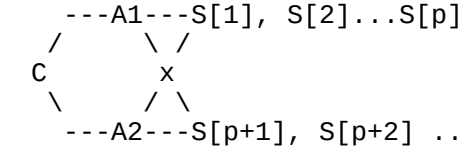

Figure 11: Shared Server Pools

A.6. Agent Chains

The scenario in Figure 12 is similar to that of Figure 8, except that, instead of the client possibly needing to select an agent that can route requests to the least loaded server, in this case A1 and A2 need to make similar decisions when selecting between A3 or A4. As the former scenario, this could be mitigated if A3 and A4 aggregate upstream loads into the load information they report downstream.

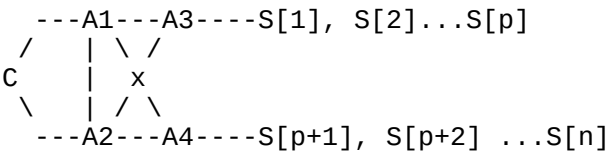

Figure 12: Agent Chains

A.7. Fully Meshed Layers

Figure 13 extends the scenario in Figure 11 by adding an extra layer of agents. But since each layer of nodes can reach any node in the next layer, each node only needs to consider the load of its next-hop peer.

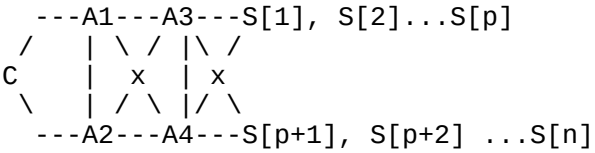

Figure 13: Full Mesh

A.8. Partitions

A Diameter network with multiple servers is said to be "partitioned" when only a subset of available servers can serve a particular realm-routed request. For example, one group of servers may handle users whose names start with "A" through "M", and another group may handle "N" through "Z".

In such a partitioned network, nodes cannot load-balance requests across partitions, since not all servers can handle the request. A client, or an intermediate agent, may still be able to load-balance between servers inside a partition.

A.9. Active-Standby Nodes

The previous scenarios assume that traffic can be load balanced among all peers that are eligible to handle a request. That is, the peers operate in an "active-active" configuration. In an "active-standby" configuration, traffic would be load-balanced among active peers. Requests would only be sent to peers in a "standby" state if the active peers became unavailable. For example, requests might be diverted to a stand-by peer if one or more active peers becomes overloaded.

A.10. Addition and removal of Nodes

When a Diameter node is added, the new node will start by advertising its load. Downstream nodes will need to factor the new load information into load balancing decisions. The downstream nodes should attempt to ensure a smooth increase of the traffic to the new node, avoiding an immediate spike of traffic to the new node. The handling of such a smooth increase is implementation specific but it can rely on the evolution of load information received from the new node and from the other nodes.

When removing a node in a controlled way (e.g. for maintenance purpose, so outside a failure case), it might be appropriate to progressively reduce the traffic to this node by routing traffic to other nodes. Simple load information (load percentage) would be not sufficient. The handling of the node removal is implementation specific but it can rely on the evolution of the load information received from the node to be removed

Authors' Addresses

Ben Campbell
Oracle
7460 Warren Parkway # 300
Frisco, Texas 75034
USA

Email: ben@nostrum.com

Steve Donovan (editor)
Oracle
7460 Warren Parkway # 300
Frisco, Texas 75034
United States

Email: srdonovan@usdonovans.com

Jean-Jacques Trottin
Nokia
Route de Villejust
91620 Nozay
France

Email: jean-jacques.trottin@nokia.com