Diameter Maintenance and Extensions (DIME)
Internet-Draft
Intended status: Standards Track
Expires: April 30, 2015

J. Korhonen, Ed.
Broadcom
S. Donovan, Ed.
B. Campbell
Oracle
L. Morand
Orange Labs

October 27, 2014

Diameter Overload Indication Conveyance
draft-ietf-dime-ovli-04.txt

Abstract

This specification documents a Diameter Overload Control (DOC) base solution and the dissemination of the overload report information.

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 30, 2015.

Table of Contents

1. Introduction

This specification defines a base solution for Diameter Overload Control (DOC), referred to as Diameter Overload Indication Conveyance (DOIC). The requirements for the solution are described and discussed in the corresponding design requirements document [RFC7068]. Note that the overload control solution defined in this specification does not address all the requirements listed in [RFC7068]. A number of overload control related features are left for the future specifications. See Appendix A for a list of extensions that are currently being considered. See Appendix C for an analysis of the conformance to the requirements specified in [RFC7068].

The solution defined in this specification addresses Diameter overload control between Diameter nodes that support the DOIC solution. Furthermore, the solution which is designed to apply to existing and future Diameter applications, requires no changes to the Diameter base protocol [RFC6733] and is deployable in environments where some Diameter nodes do not implement the Diameter overload control solution defined in this specification.

2. Terminology and Abbreviations

Abatement

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

Abatement Algorithm

An mechanism requested by reporting nodes and used by reacting nodes to reduce the amount of traffic sent during an occurrence of overload control.

Diversion

Abatement of traffic sent to a reporting node by a reacting node in response to receipt of an overload report. The abatement is achieved by diverting traffic from the reporting node to another Diameter node that is able to process the request.

Host-Routed Request

The set of requests that a reacting node knows will be served by a particular host, either due to the presence of a Destination-Host AVP, or by some other local knowledge on the part of the reacting node.

Overload Control State (OCS)

Reporting and reacting node internally maintained state describing occurrences of overload control.

Overload Report (OLR)

Information sent by a reporting node indicating the start, continuation or end of an occurrence of overload control.

Reacting Node

A Diameter node that acts upon an overload report.

Realm-Routed Request

The set of requests that a reacting node does not know the host that will service the request.

Reporting Node

A Diameter node that generates an overload report. (This may or may not be the overloaded node.)

Throttling

Throttling is the reduction of the number of requests sent to an entity. Throttling can include a Diameter Client or Diameter Server dropping requests, or a Diameter Agent rejecting requests with appropriate error responses. In extreme cases reporting nodes can also throttle requests when the requested reductions in traffic does not sufficiently address the overload scenario.

3. Solution Overview

The Diameter Overload Information Conveyance (DOIC) solution allows Diameter nodes to request other nodes to perform overload abatement actions, that is, actions to reduce the load offered to the overloaded node or realm.

A Diameter node that supports DOIC is known as a "DOIC node". Any Diameter node can act as a DOIC node, including clients, servers, and agents. DOIC nodes are further divided into "Reporting Nodes" and "Reacting Nodes." A reporting node requests overload abatement by sending an Overload Report (OLR) to one or more reacting nodes.

A reacting node acts upon OLRs, and performs whatever actions are needed to fulfil the abatement requests included in the OLRs. A Reporting node may report overload on its own behalf, or on behalf of other (typically upstream) nodes. Likewise, a reacting node may perform overload abatement on its own behalf, or on behalf of other (typically downstream) nodes.

A node's role as a DOIC node is independent of its Diameter role. For example, Diameter Relay and Proxy Agents may act as DOIC nodes, even though they are not endpoints in the Diameter sense. Since Diameter enables bi-directional applications, where Diameter Servers can send requests towards Diameter Clients, a given Diameter node can simultaneously act as a reporting node and a reacting node.

Likewise, a relay or proxy agent may act as a reacting node from the perspective of upstream nodes, and a reporting node from the perspective of downstream nodes.

DOIC nodes do not generate new messages to carry DOIC related information. Rather, they "piggyback" DOIC information over existing Diameter messages by inserting new AVPs into existing Diameter requests and responses. Nodes indicate support for DOIC, and any needed DOIC parameters by inserting an OC_Supported_Features AVP

(Section 6.2) into existing requests and responses. Reporting nodes send OLRs by inserting OC-OLR AVPs (Section 6.3).

A given OLR applies to the Diameter realm and application of the Diameter message that carries it. If a reporting node supports more than one realm and/or application, it reports independently for each combination of realm and application. Similarly, the OC-Supported-Features AVP applies to the realm and application of the enclosing message. This implies that a node may support DOIC for one application and/or realm, but not another, and may indicate different DOIC parameters for each application and realm for which it supports DOIC.

Reacting nodes perform overload abatement according to an agreed-upon abatement algorithm. An abatement algorithm defines the meaning of the parameters of an OLR and the procedures required for overload abatement. This document specifies a single must-support algorithm, namely the "loss" algorithm (Section 5). Future specifications may introduce new algorithms.

Overload conditions may vary in scope. For example, a single Diameter node may be overloaded, in which case reacting nodes may reasonably attempt to send requests to other destinations or via other agents. On the other hand, an entire Diameter realm may be overloaded, in which case such attempts would do harm. DOIC OLRs have a concept of "report type" (Section 6.6), where the type defines such behaviors. Report types are extensible. This document defines report types for overload of a specific server, and for overload of an entire realm.

A report of type host is sent to indicate the overload of a specific server for the application-id indicated in the transaction. When receiving an OLR of type host, a reacting node applies overload abatement to what is referred to in this document as host-routed requests. This is the set of requests that the reacting node knows will be served by a particular host, either due to the presence of a Destination-Host AVP, or by some other local knowledge on the part of the reacting node. The reacting node applies overload abatement on those host-routed requests which the reacting node knows will be served by the server that matches the Origin-Host AVP of the received message that contained the received OLR of type host.

A report type of realm is sent to indicate the overload of all servers in a realm for the application-id. When receiving an OLR of type realm, a reacting node applies overload abatement to what is referred to in this document as realm-routed requests. This is the set of requests that are not host-routed as defined in the previous paragraph.

While a reporting node sends OLRs to "adjacent" reacting nodes, nodes that are "adjacent" for DOIC purposes may not be adjacent from a Diameter, or transport, perspective. For example, one or more Diameter agents that do not support DOIC may exist between a given pair of reporting and reacting nodes, as long as those agents pass unknown AVPs through unchanged. The report types described in this document can safely pass through non-supporting agents. This may not be true for report types defined in future specifications. Documents that introduce new report types MUST describe any limitations on their use across non-supporting agents.

3.1. Piggybacking Principle

The overload control AVPs defined in this specification have been designed to be piggybacked on top of existing application messages. This is made possible by adding overload control top-level AVPs, the OC-OLR AVP and the OC-Supported-Features AVP, as optional AVPs into existing commands when the corresponding Command Code Format (CCF) specification allows adding new optional AVPs (see Section 1.3.4 of [RFC6733]).

Reacting nodes indicate support for DOIC by including the OC-Supported-Features AVP in all request messages originated or relayed by the reacting node.

Reporting nodes indicate support for DOIC by including the OC-Supported-Features AVP in all answer messages originated or relayed by the reporting node. Reporting nodes also include overload reports using the OC-OLR AVP in answer messages.

> Note: There is no new Diameter application defined to carry overload related AVPs. The DOIC AVPs are carried in existing Diameter application messages.

Note that the overload control solution does not have fixed server and client roles. The DOIC node role is determined based on the message type: whether the message is a request (i.e. sent by a "reacting node") or an answer (i.e. send by a "reporting node"). Therefore, in a typical "client-server" deployment, the Diameter Client MAY report its overload condition to the Diameter Server for any Diameter Server initiated message exchange. An example of such is the Diameter Server requesting a re-authentication from a Diameter Client.

3.2. DOIC Capability Announcement

The DOIC solution supports the ability for Diameter nodes to determine if other nodes in the path of a request support the solution. This capability is referred to as DOIC Capability Announcement (DCA) and is separate from Diameter Capability Exchange.

The DCA solution uses the OC-Supported-Features AVPs to indicate the Diameter overload features supported.

The first node in the path of a Diameter request that supports the DOIC solution inserts the OC-Supported-Feature AVP in the request message. This includes an indication that it supports the loss overload abatement algorithm defined in this specification (see Section 5). This ensures that there is at least one commonly supported overload abatement algorithm between the reporting node and the reacting nodes in the path of the request.

> DOIC must support deployments where Diameter Clients and/or Diameter Servers do not support the DOIC solution. In this scenario, it is assumed that Diameter Agents that support the DOIC solution will handle overload abatement for the non supporting Diameter nodes. In this case the DOIC agent will insert the OC-Supporting-Features AVP in requests that do not already contain one, telling the reporting node that there is a DOIC node that will handle overload abatement.

The reporting node inserts the OC-Supported-Feature AVP in all answer messages to requests that contained the OC-Supported-Feature AVP. The contents of the reporting node's OC-Supported-Feature AVP indicate the set of Diameter overload features supported by the reporting node with one exception.

The reporting node only includes an indication of support for one overload abatement algorithm. This is the algorithm that the reporting node intends to use should it enter an overload condition or requests to use while it actually is in an overload condition. Reacting nodes can use the indicated overload abatement algorithm to prepare for possible overload reports and must use the indicated overload abatement algorithm if traffic reduction is actually requested.

> Note that the loss algorithm defined in this document is a stateless abatement algorithm. As a result it does not require any actions by reacting nodes prior to the receipt of an overload report. Stateful abatement algorithms that base the abatement logic on a history of request messages sent might require reacting

nodes to maintain state to ensure that overload reports can be properly handled.

The individual features supported by the DOIC nodes are indicated in the OC-Feature-Vector AVP. Any semantics associated with the features will be defined in extension specifications that introduce the features.

The DCA mechanism must also support the scenario where the set of features supported by the sender of a request and by agents in the path of a request differ. In this case, the agent updates the OC-Supported-Feature AVP to reflect the mixture of the two sets of supported features.

The logic to determine the content of the modified OC-Supported-Feature AVP is out-of-scope for this specification and is left to implementation decisions. Care must be taken not to introduce interoperability issues for downstream or upstream DOIC nodes.

3.3. DOIC Overload Condition Reporting

As with DOIC Capability Announcement, Overload Condition Reporting uses new AVPs (Section 6.3) to indicate an overload condition.

The OC-OLR AVP is referred to as an overload report. The OC-OLR AVP includes the type of report, a sequence number, the length of time that the report is valid and abatement algorithm specific AVPs.

Two types of overload reports are defined in this document, host reports and realm reports.

A report of type host is sent to indicate the overload of a specific Diameter node for the application-id indicated in the transaction. When receiving an OLR of type host, a reacting node applies overload abatement to what is referred to in this document as host-routed requests. This is the set of requests that the reacting node knows will be served by a particular host, either due to the presence of a Destination-Host AVP, or by some other local knowledge on the part of the reacting node. The reacting node applies overload abatement on those host-routed requests which the reacting node knows will be served by the server that matches the Origin-Host AVP of the received message that contained the received OLR of type host.

Realm reports apply to realm-routed requests for a specific realm as indicated in the Destination-Realm AVP.

Reporting nodes are responsible for determining the need for a reduction of traffic. The method for making this determination is

implementation specific and depend on the type of overload report being generated. A host report, for instance, will generally be generated by tracking utilization of resources required by the host to handle transactions for the Diameter application. A realm report will generally impact the traffic sent to multiple hosts and, as such, will typically require tracking the capacity of the servers able to handle realm-routed requests for the application.

Once a reporting node determines the need for a reduction in traffic, it uses the DOIC defined AVPs to report on the condition. These AVPs are included in answer messages sent or relayed by the reporting node. The reporting node indicates the overload abatement algorithm that is to be used to handle the traffic reduction in the OC-Supported-Features AVP. The OC-OLR AVP is used to communicate information about the requested reduction.

Reacting nodes, upon receipt of an overload report, are responsible for applying the abatement algorithm to traffic impacted by the overload report. The method used for that abatement is dependent on the abatement algorithm. The loss abatement algorithm is defined in this document (Section 5). Other abatement algorithms can be defined in extensions to the DOIC solutions.

As the conditions that lead to the generation of the overload report change the reporting node can send new overload reports requesting greater reduction if the condition gets worse or less reduction if the condition improves. The reporting node sends an overload report with a duration of zero to indicate that the overload condition has ended and use of the abatement algorithm is no longer needed.

The reacting node also determines when the overload report expires based on the OC-Validity-Duration AVP in the overload report and stops applying the abatement algorithm when the report expires.

3.4. DOIC Extensibility

The DOIC solution is designed to be extensible. This extensibility is based on existing Diameter based extensibility mechanisms.

There are multiple categories of extensions that are expected. This includes the definition of new overload abatement algorithms, the definition of new report types and new definitions of the scope of messages impacted by an overload report.

The DOIC solution uses the OC-Supported-Features AVP for DOIC nodes to communicate supported features. The specific features supported by the DOIC node are indicated in the OC-Feature-Vector AVP. DOIC extensions must define new values for the OC-Feature-Vector AVP.

DOIC extensions also have the ability to add new AVPs to the OC-Supported-Features AVP, if additional information about the new feature is required.

Reporting nodes use the OC-OLR AVP to communicate overload occurrences. This AVP can also be extended to add new AVPs allowing a reporting nodes to communicate additional information about handling an overload condition.

If necessary, new extensions can also define new top-level AVPs. It is, however, recommended that DOIC extensions use the OC-Supported-Features and OC-OLR to carry all DOIC related AVPs.

3.5. Simplified Example Architecture

Figure 1 illustrates the simplified architecture for Diameter overload information conveyance.

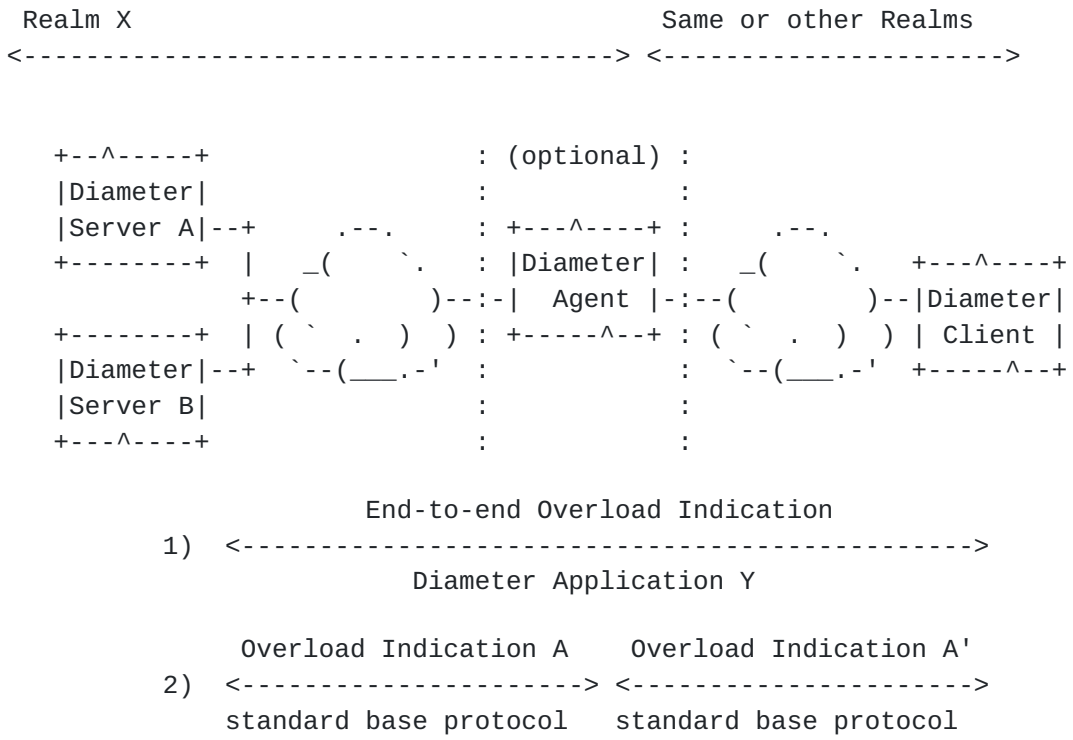

Figure 1: Simplified architecture choices for overload indication delivery

In Figure 1, the Diameter overload indication can be conveyed (1) end-to-end between servers and clients or (2) between servers and

Diameter agent inside the realm and then between the Diameter agent and the clients.

4. Solution Procedures

This section outlines the normative behavior associated with the DOIC solution.

4.1. Capability Announcement

This section defines DOIC Capability Announcement (DCA) behavior.

4.1.1. Reacting Node Behavior

A reacting node MUST include the OC-Supported-Features AVP in all request messages.

A reacting node MAY include the OC-Feature-Vector AVP with an indication of the loss algorithm. A reacting node MUST include the OC-Feature-Vector AVP to indicate support for abatement algorithms in addition to the loss algorithm.

A reacting node SHOULD indicate support for all other DOIC features it supports.

> Not all DOIC features will necessarily apply to all transactions. For instance, there may be a future extension that only applies to session based applications. A reacting node that supports this extension can choose to not include it for non session based applications.

An OC-Supported-Features AVP in answer messages indicates there is a reporting node for the transaction. The reacting node MAY take action based on the features indicated in the OC-Feature-Vector AVP.

> Note that the loss abatement algorithm is the only feature described in this document and it does not require action to be taken when there is an active overload report. This behavior is described in Section 4.2 and Section 5.

4.1.2. Reporting Node Behavior

Upon receipt of a request message, a reporting node determines if there is a reacting node for the transaction based on the presence of the OC-Supported-Features AVP.

If the request message contains an OC-Supported-Features AVP then the reporting node MUST include the OC-Supported-Features AVP in the answer message for that transaction.

The reporting node MUST NOT include the OC-Supported-Features AVP, OC-OLR AVP or any other overload control AVPs defined in extension drafts in response messages for transactions where the request message does not include the OC-Supported-Features AVP. Lack of the OC-Supported-Features AVP in the request message indicates that there is no reacting node for the transaction.

Based on the content of the OC-Supported-Features AVP in the request message, the reporting node knows what overload control functionality is supported by the reacting node. The reporting node then acts accordingly for the subsequent answer messages it initiates.

The reporting node MUST indicate support for one and only one abatement algorithm in the OC-Feature-Vector AVP. The abatement algorithm included MUST be from the set of abatement algorithms contained in the request message's OC-Supported-Features AVP. The abatement algorithm included MUST indicate the abatement algorithm the reporting node wants the reacting node to use when the reporting node enters an overload condition.

For an ongoing overload state, a reacting node MUST keep the algorithm that was selected by the reporting node in further requests towards the reporting node. The reporting node SHOULD NOT change the selected algorithm during a period of time that it is in an overload condition and, as a result, is sending OC-OLR AVPs in answer messages.

The reporting node SHOULD indicate support for other DOIC features defined in extension drafts that it supports and that apply to the transaction.

> Note that not all DOIC features will apply to all Diameter applications or deployment scenarios. The features included in the OC-Feature-Vector AVP are based on local reporting node policy.

4.1.3. Agent Behavior

Diameter agents that support DOIC MUST ensure that all messages have the OC-Supporting-Features AVP. If a message handled by the DOIC agent does not include the OC-Supported-Features AVP then the DOIC agent inserts the AVP. If the message already has the AVP then the agent either leaves it unchanged in the relayed message or modifies it to reflect a mixed set of DOIC features.

An agent MAY modify the OC-Supported-Features AVP carried in answer messages.

For instance, if the agent supports a superset of the features reported by the reacting node then the agent might choose, based on local policy, to advertise that superset of features to the reporting node.

If the agent modifies the OC-Supported-Features AVP sent to the reporting node then it might also need to modify the OC-Supported-Features AVP sent to a reacting node in the subsequent answer message, as it cannot send an indication of support for features that are not supported by the reacting node.

Editor's note: There is an open issue on the wording around agent behavior in this case that needs to be resolved prior to finishing this document.

4.2. Overload Report Processing

4.2.1. Overload Control State

Both reacting and reporting nodes maintain Overload Control State (OCS) for active overload conditions.

4.2.1.1. Overload Control State for Reacting Nodes

A reacting node SHOULD maintain the following OCS per supported Diameter application:

- A host-type OCS entry for each Destination-Host to which it sends host-type requests and
- A realm-type OCS entry for each Destination-Realm to which it sends realm-type requests.

A host-type OCS entry is identified by the pair of Application-Id and Host-Id.

A realm-type OCS entry is identified by the pair of Application-Id and Realm-Id.

The host-type and realm-type OCS entries MAY include the following information (the actual information stored is an implementation decision):

- Sequence number (as received in OC-OLR)

- 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 OC-Supported-Features AVP)

- Abatement Algorithm specific input data (as received within the OC-OLR AVP, for example, OC-Reduction-Percentage for the Loss abatement algorithm)

4.2.1.2. Overload Control State for Reporting Nodes

A reporting node SHOULD maintain OCS entries per supported Diameter application, per supported (and eventually selected) Abatement Algorithm and per report-type.

An OCS entry is identified by the pair of Application-Id and Abatement Algorithm.

The OCS entry for a given pair of Application and Abatement Algorithm MAY include the information (the actual information stored is an implementation decision):

- Report type

- Sequence number

- Validity Duration

- Expiration Time

- Algorithm specific input data (for example, the Reduction Percentage for the Loss Abatement Algorithm)

4.2.1.3. Reacting Node Maintenance of Overload Control State

When a reacting node receives an OC-OLR AVP, it MUST determine if it is for an existing or new overload condition.

> For the remainder of this section the term OLR referres to the combination of the contents of the received OC-OLR AVP and the abatement algorithm indicated in the received OC-Supported-Features AVP.

The OLR is for an existing overload condition if the reacting node has an OCS that matches the received OLR.

For a host report-type this means it matches the app-id and host-id in an existing host OCS entry.

For a realm report-type this means it matches the app-id and realm-id in an existing realm 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 host report-type this means it creates on OCS entry with the app-id of the application-id in the received message and host-id of the Origin-Host in the received message.

> Note: This solution assumes that the Origin-Host AVP in the answer message included by the reporting node is not changed along the path to the reacting node.

For a realm report-type this means it creates on OCS entry with the app-id of the application-id in the received message and realm-id of the Origin-Realm in the received message.

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

> Note that it is not necessarily appropriate to delete the OCS entry, as there is recommended behavior that the reacting node slowly returns to full traffic when ending an overload abatement period.

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").

4.2.1.4. Reporting Node Maintenance of Overload Control State

A reporting node SHOULD create a new OCS entry when entering an overload condition.

> If the reporting node knows through absence of the OC-Supported-Features AVP in received messages that there are no reacting nodes supporting DOIC then the reporting node can choose to not create OCS entries.

When generating a new OCS entry the sequence number MAY be set to any value if there is no unexpired overload report for previous overload conditions sent to any reacting node for the same application and report-type.

When generating sequence numbers for new overload conditions, the new sequence number MUST be greater than any sequence number in an active (unexpired) overload report previously sent by the reporting node. This property MUST hold over a reboot of the reporting node.

The reporting node MUST update an OCS entry when it needs to adjust the validity duration of the overload condition at reacting nodes.

> For instance, if the reporting node wishes to instruct reacting nodes to continue overload abatement for a longer period of time that originally communicated. This also applies if the reporting node wishes to shorten the period of time that overload abatement is to continue.

A reporting node MUST NOT update the abatement algorithm in an active OCS entry.

A reporting node MUST update an OCS entry when it wishes to adjust any abatement algorithm specific parameters, including the reduction percentage used for the Loss abatement algorithm.

> For instance, if the reporting node wishes to change the reduction percentage either higher, if the overload condition has worsened, or lower, if the overload condition has improved, then the reporting node would update the appropriate OCS entry.

The reporting node MUST update the sequence number associated with the OCS entry anytime the contents of the OCS entry are changed. This will result in a new sequence number being sent to reacting nodes, instructing the reacting nodes to process the OC-OLR AVP.

A reporting node SHOULD update an OCS entry with a validity duration of zero ("0") when the overload condition ends.

If the reporting node knows that the OCS entries in the reacting nodes are near expiration then the reporting node can decide to delete the OCS entry.

The reporting node MUST keep an OCS entry with a validity duration of zero ("0") for a period of time long enough to ensure that any non-expired reacting node's OCS entry created as a result of the overload condition in the reporting node is deleted.

4.2.2. Reacting Node Behavior

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

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

For the Loss abatement algorithm defined in this specification, see Section 5 for the abatement logic applied.

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 section Section 7.

In the case that the OCS entry validity duration expires or has a validity duration of zero ("0"), meaning that it 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.

4.2.3. Reporting Node Behavior

The operation on the reporting node is straight forward.

If there is an active OCS entry then the reporting node SHOULD include the OC-OLR AVP in all answer messages to requests that contain the OC-Supported-Features AVP and that match the active OCS entry.

A request matches if the application-id in the request matches the application-id in any active OCS entry and if the report-type in the OCS entry matches a report-type supported by the reporting node as indicated in the OC-Supported-Features AVP.

The contents of the OC-OLR AVP MUST contain all information necessary for the abatement algorithm indicated in the OC-Supported-Features AVP that is also included in the answer message.

A reporting node MAY choose to not resend an overload report to a reacting node if it can guarantee that this overload report is already active in the reacting node.

Note - In some cases (e.g. when there are one or more agents in the path between reporting and reacting nodes, or when overload reports are discarded by reacting nodes) the reporting node may not be able to guarantee that the reacting node has received the report.

A reporting node MUST NOT send overload reports of a type that has not been advertised as supported by the reacting node.

Note that a reacting node advertises support for the host and realm report types by including the OC-Supported-Features AVP in the request. Support for other report types must be explicitly indicated by new feature bits in the OC-Feature-Vector AVP.

A reporting node MAY rely on the OC-Validity-Duration AVP values for the implicit overload control state cleanup on the reacting node. However, it is RECOMMENDED that the reporting node always explicitly indicates the end of a overload condition.

The reporting node SHOULD indicate the end of an overload occurrence by sending a new OLR with OC-Validity-Duration set to a value of zero ("0"). The reporting node SHOULD ensure that all reacting nodes receive the updated overload report.

All OLRs sent have an expiration time calculated by adding the validity-duration contained in the OLR to the time the message was sent. Transit time for the OLR can be safely ignored. The reporting node can ensure that all reacting nodes have received the OLR by continuing to send it in answer messages until the expiration time for all OLRs sent for that overload condition have expired.

When a reporting node sends an OLR, it effectively delegates any necessary throttling to downstream nodes. Therefore, the reporting node SHOULD NOT apply throttling to the set of messages to which the OLR applies. That is, the same candidate set of messages SHOULD NOT be throttled multiple times.

However, when the reporting node sends and OLR downstream, it MAY still be responsible to apply other abatement methods such as diversion. The reporting node might also need to throttle requests for reasons other then overload. For example, an agent or server might have a configured rate limit for each client, and throttle

requests that exceed that limit, even if such requests had already been candidates for throttling by downstream nodes.

This document assumes that there is a single source for realm-reports for a given realm, or that if multiple nodes can send realm reports, that each such node has full knowledge of the overload state of the entire realm. A reacting node cannot distinguish between receiving realm-reports from a single node, or from multiple nodes.

> Editor's Note: There is not yet consensus on the above two paragraphs. Two alternatives are under consideration -- synchronization of sequence numbers and attribution of reports. If no consensus is reached then it will be left to be addressed as an extension.

4.3. Protocol Extensibility

The overload control solution can be extended, e.g. with new traffic abatement algorithms, new report types or other new functionality.

When defining a new extension a new feature bit MUST be defined for the OC-Feature-Vector. This feature bit is used to communicate support for the new feature.

The extension MAY define new AVPs for use in DOIC Capability Announcement and for use in DOIC Overload reporting. These new AVPs SHOULD be defined to be extensions to the OC-Supported-Features and OC-OLR AVPs defined in this document.

It should be noted that [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.

The handling of feature bits in the OC-Feature-Vector AVP that are not associated with overload abatement algorithms MUST be specified by the extensions that define the features.

When defining new report type values, the corresponding specification MUST define the semantics of the new report types and how they affect the OC-OLR AVP handling. The specification MUST also reserve a corresponding new feature bit in the OC-Feature-Vector AVP.

The OC-OLR AVP can be expanded with optional sub-AVPs only if a legacy DOIC implementation can safely ignore them without breaking backward compatibility for the given OC-Report-Type AVP value. If the new sub-AVPs imply new semantics for handling the indicated report type, then a new OC-Report-Type AVP value MUST be defined.

New features (feature bits in the OC-Feature-Vector AVP) and report types (in the OC-Report-Type AVP) MUST be registered with IANA. As with any Diameter specification, new AVPs MUST also be registered with IANA. See Section 8 for the required procedures.

5. Loss Algorithm

This section documents the Diameter overload loss abatement algorithm.

5.1. Overview

The DOIC specification supports the ability for multiple overload abatement algorithms to be specified. The abatement algorithm used for any instance of overload is determined by the Diameter Overload Capability Announcement process documented in Section 4.1.

The loss algorithm described in this section is the default algorithm that must be supported by all Diameter nodes that support DOIC.

The loss algorithm is designed to be a straightforward and stateless overload abatement algorithm. It is used by reporting nodes to request a percentage reduction in the amount of traffic sent. The traffic impacted by the requested reduction depends on the type of overload report.

Reporting nodes use a strategy of applying abatement logic to the requested percentage of request messages sent (or handled in the case of agents) by the reacting node that are impacted by the overload report.

From a conceptual level, the logic at the reacting node could be outlined as follows.

1. An overload report is received and the associated overload state is either saved or updated (if required) by the reacting node.

2. A new Diameter request is generated by the application running on the reacting node.

3. The reacting node determines that an active overload report applies to the request, as indicated by the corresponding OCS entry.

4. The reacting node determines if abatement should be applied to the request. One approach that could be taken for each request is to select a random number between 1 and 100. If the random number is less than the indicated reduction percentage then the

request is given abatement treatment, otherwise the request is given normal routing treatment.

5.2. Reporting Node Behavior

The method a reporting nodes uses to determine the amount of traffic reduction required to address an overload condition is an implementation decision.

When a reporting node that has selected the loss abatement algorithm determines the need to request a traffic reduction it includes an OC-OLR AVP in response messages as described in Section 4.2.3.

The reporting node MUST indicate a percentage reduction in the OC-Reduction-Percentage AVP.

The reporting node MAY change the reduction percentage in subsequent overload reports. When doing so the reporting node must conform to overload report handing specified in Section 4.2.3.

When the reporting node determines it no longer needs a reduction in traffic the reporting node SHOULD send an overload report indicating the overload report is no longer valid, as specified in Section 4.2.3.

5.3. Reacting Node Behavior

The method a reacting node uses to determine which request messages are given abatement treatment is an implementation decision.

When receiving an OC-OLR in an answer message where the algorithm indicated in the OC-Supported-Features AVP is the loss algorithm, the reacting node MUST apply abatement treatment to the requested percentage of request messages sent.

Note: the loss algorithm is a stateless algorithm. As a result, the reacting node does not guarantee that there will be an absolute reduction in traffic sent. Rather, it guarantees that the requested percentage of new requests will be given abatement treatment.

When applying overload abatement treatment for the load abatement algorithm, the reacting node MUST abate, either by throttling or diversion, the requested percentage of requests that would have otherwise been sent to the reporting host or realm.

If reacting node comes out of the 100 percent traffic reduction as a result of the overload report timing out, the following concerns are

RECOMMENDED to be applied. The reacting node sending the traffic should be conservative and, for example, first send "probe" messages to learn the overload condition of the overloaded node before converging to any traffic amount/rate decided by the sender. Similar concerns apply in all cases when the overload report times out unless the previous overload report stated 0 percent reduction.

If the reacting node does not receive an OLR in messages sent to the formerly overloaded node then the reacting node SHOULD slowly increase the rate of traffic sent to the overloaded node.

It is suggested that the reacting node decrease the amount of traffic given abatement treatment by 20% each second until the reduction is completely removed and no traffic is given abatement treatment.

> The goal of this behavior is to reduce the probability of overload condition thrashing where an immediate transition from 100% reduction to 0% reduction results in the reporting node moving quickly back into an overload condition.

6. Attribute Value Pairs

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

A new application specification can incorporate the overload control mechanism specified in this document by making it mandatory to implement for the application and referencing this specification normatively. It is the responsibility of the Diameter application designers to define how overload control mechanisms works on that application.

6.1. OC-Supported-Features AVP

The OC-Supported-Features AVP (AVP code TBD1) is type of Grouped and serves two purposes. First, it announces a node's support for the DOIC solution in general. Second, it contains the description of the supported DOIC features of the sending node. The OC-Supported-Features AVP MUST be included in every Diameter request message a DOIC supporting node sends.

```
OC-Supported-Features ::= < AVP Header: TBD1 >
                            [ OC-Feature-Vector ]
                          * [ AVP ]
```

The OC-Feature-Vector sub-AVP is used to announce the DOIC features supported by the DOIC node, in the form of a flag bits field in which each bit announces one feature or capability supported by the node (see Section 6.2). The absence of the OC-Feature-Vector AVP indicates that only the default traffic abatement algorithm described in this specification is supported.

6.2. OC-Feature-Vector AVP

The OC-Feature-Vector AVP (AVP code TBD6) is type of Unsigned64 and contains a 64 bit flags field of announced capabilities of a DOIC node. The value of zero (0) is reserved.

The following capabilities are defined in this document:

OLR_DEFAULT_ALGO (0x0000000000000001)

When this flag is set by the DOIC node it means that the default traffic abatement (loss) algorithm is supported.

6.3. OC-OLR AVP

The OC-OLR AVP (AVP code TBD2) is type of Grouped and contains the information necessary to convey an overload report on an overload condition at the reporting node. The OC-OLR AVP does not explicitly contain all information needed by the reacting node to decide whether a subsequent request must undergo a throttling process with the received reduction percentage. The value of the OC-Report-Type AVP within the OC-OLR AVP indicates which implicit information is relevant for this decision (see Section 6.6). The application the OC-OLR AVP applies to is the same as the Application-Id found in the Diameter message header. The host or realm the OC-OLR AVP concerns is determined from the Origin-Host AVP and/or Origin-Realm AVP found in the encapsulating Diameter command. The OC-OLR AVP is intended to be sent only by a reporting node.

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

Note that if a Diameter command were to contain multiple OC-OLR AVPs they all MUST have different OC-Report-Type AVP value. OC-OLR AVPs with unknown values SHOULD be silently discarded by reacting nodes and the event SHOULD be logged.

6.4. OC-Sequence-Number AVP

The OC-Sequence-Number AVP (AVP code TBD3) is type of Unsigned64. Its usage in the context of overload control is described in Section 4.2.

From the functionality point of view, the OC-Sequence-Number AVP MUST be used as a non-volatile increasing counter for a sequence of overload reports between two DOIC nodes for the same overload occurrence. The sequence number is only required to be unique between two DOIC nodes. Sequence numbers are treated in a uni-directional manner, i.e. two sequence numbers on each direction between two DOIC nodes are not related or correlated.

6.5. OC-Validity-Duration AVP

The OC-Validity-Duration AVP (AVP code TBD4) is type of Unsigned32 and indicates in milliseconds the validity time of the overload report. The number of milliseconds is measured after reception of the first OC-OLR AVP with a given value of OC-Sequence-Number AVP. The default value for the OC-Validity-Duration AVP is 5000 (i.e., 5 seconds). When the OC-Validity-Duration AVP is not present in the OC-OLR AVP, the default value applies. Validity duration with values above 86400 (i.e.; 24 hours) MUST NOT be used. Invalid duration values are treated as if the OC-Validity-Duration AVP were not present and result in the default value being used.

Editor's note: There is an open discussion on whether to have an upper limit on the OC-Validity-Duration value, beyond that which can be indicated by an Unsigned32.

A timeout of the overload report has specific concerns that need to be taken into account by the DOIC node acting on the earlier received overload report(s). Section 6.7 discusses the impacts of timeout in the scope of the traffic abatement algorithms.

6.6. OC-Report-Type AVP

The OC-Report-Type AVP (AVP code TBD5) is type of Enumerated. The value of the AVP describes what the overload report concerns. The following values are initially defined:

0 A host report. The overload treatment should apply to requests for which all of the following conditions are true:

  Either the Destination-Host AVP is present in the request and its value matches the value of the Origin-Host AVP of the received message that contained the OC-OLR AVP; or the Destination-Host is

not present in the request but the value of the peer identity associated with the connection used to send the request matches the value of the Origin-Host AVP of the received message that contained the OC-OLR AVP.

The value of the Destination-Realm AVP in the request matches the value of the Origin-Realm AVP of the received message that contained the OC-OLR AVP.

The value of the Application-ID in the Diameter Header of the request matches the value of the Application-ID of the Diameter Header of the received message that contained the OC-OLR AVP.

1 A realm report. The overload treatment should apply to requests for which all of the following conditions are true:

The Destination-Host AVP is absent in the requestand the value of the peer identity associated with the connection used to send the request does not match a server that could serve the request.

The value of the Destination-Realm AVP in the request matches the value of the Origin-Realm AVP of the received message that contained the OC-OLR AVP.

The value of the Application-ID in the Diameter Header of the request matches the value of the Application-ID of the Diameter Header of the received message that contained the OC-OLR AVP.

The OC-Report-Type AVP is envisioned to be useful for situations where a reacting node needs to apply different overload treatments for different overload contexts. For example, the reacting node(s) might need to throttle differently requests sent to a specific server (identified by the Destination-Host AVP in the request) and requests that can be handled by any server in a realm.

6.7. OC-Reduction-Percentage AVP

The OC-Reduction-Percentage AVP (AVP code TBD8) is type of Unsigned32 and describes the percentage of the traffic that the sender is requested to reduce, compared to what it otherwise would send. The OC-Reduction-Percentage AVP applies to the default (loss) algorithm specified in this specification. However, the AVP can be reused for future abatement algorithms, if its semantics fit into the new algorithm.

The value of the Reduction-Percentage AVP is between zero (0) and one hundred (100). Values greater than 100 are ignored. The value of 100 means that all traffic is to be throttled, i.e. the reporting

node is under a severe load and ceases to process any new messages. The value of 0 means that the reporting node is in a stable state and has no need for the reacting node to apply any traffic abatement. The default value of the OC-Reduction-Percentage AVP is 0. When the OC-Reduction-Percentage AVP is not present in the overload report, the default value applies.

6.8. Attribute Value Pair flag rules

Attribute Name	AVP Code	Section Defined	Value Type	AVP flag rules: MUST	AVP flag rules: MUST NOT
OC-Supported-Features	TBD1	x.x	Grouped		V
OC-OLR	TBD2	x.x	Grouped		V
OC-Sequence-Number	TBD3	x.x	Unsigned64		V
OC-Validity-Duration	TBD4	x.x	Unsigned32		V
OC-Report-Type	TBD5	x.x	Enumerated		V
OC-Reduction -Percentage	TBD8	x.x	Unsigned32		V
OC-Feature-Vector	TBD6	x.x	Unsigned64		V

As described in the Diameter base protocol [RFC6733], the M-bit setting for a given AVP is relevant to an application and each command within that application that includes the AVP.

The Diameter overload control AVPs SHOULD always be sent with the M-bit cleared when used within existing Diameter applications to avoid backward compatibility issues. Otherwise, when reused in newly defined Diameter applications, the DOC related AVPs SHOULD have the M-bit set.

7. Error Response Codes

When a DOIC node rejects a Diameter request due to overload, the DOIC node MUST select an appropriate error response code. This

determination is made based on the probability of the request succeeding if retried on a different path.

A reporting node rejecting a Diameter request due to an overload condition SHOULD send a DIAMETER-TOO-BUSY error response, if it can assume that the same request may succeed on a different path.

If a reporting node knows or assumes that the same request will not succeed on a different path, DIAMETER_UNABLE_TO_COMPLY error response SHOULD be used. Retrying would consume valuable resources during an occurrence of overload.

For instance, if the request arrived at the reporting node without a Destination-Host AVP then the reporting node might determine that there is an alternative Diameter node that could successfully process the request and that retrying the transaction would not negatively impact the reporting node. DIAMETER_TOO_BUSY would be sent in this case.

For instance, if the request arrived at the reporting node with a Destination-Host AVP populated with its own Diameter identity then the reporting node can assume that retrying the request would result in it coming to the same reporting node. DIAMETER_UNABLE_TO_COMPLY would be sent in this case.

A second example is when an agent that supports the DOIC solution is performing the role of a reacting node for a non supporting client. Requests that are rejected as a result of DOIC throttling by the agent in this scenario would generally be rejected with a DIAMETER_UNABLE_TO_COMPLY response code.

8. IANA Considerations

8.1. AVP codes

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

8.2. New registries

Two new registries are needed under the 'Authentication, Authorization, and Accounting (AAA) Parameters' registry.

Section 6.2 defines a new "Overload Control Feature Vector" registry including the initial assignments. New values can be added into the registry using the Specification Required policy [RFC5226]. See Section 6.2 for the initial assignment in the registry.

Section 6.6 defines a new "Overload Report Type" registry with its initial assignments. New types can be added using the Specification Required policy [RFC5226].

9. Security Considerations

This mechanism gives Diameter nodes the ability to request that downstream nodes send fewer Diameter requests. Nodes do this by exchanging overload reports that directly affect this reduction. This exchange is potentially subject to multiple methods of attack, and has the potential to be used as a Denial-of-Service (DoS) attack vector.

Overload reports may contain information about the topology and current status of a Diameter network. This information is potentially sensitive. Network operators may wish to control disclosure of overload reports to unauthorized parties to avoid its use for competitive intelligence or to target attacks.

Diameter does not include features to provide end-to-end authentication, integrity protection, or confidentiality. This may cause complications when sending overload reports between non-adjacent nodes.

9.1. 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. TCP or SCTP) connection, or the messages may traverse one or more intermediaries, known as Diameter Agents. Diameter nodes use TLS, 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 hop-by-hop 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 an overload report.

There are several ways an attacker might attempt to exploit the overload control mechanism. An unauthorized third party might inject an overload report into the network. If this third party is upstream

of an agent, and that agent fails to apply proper authorization policies, downstream nodes may mistakenly trust the report. This attack is at least partially mitigated by the assumption that nodes include overload reports in Diameter answers but not in requests. This requires an attacker to have knowledge of the original request in order to construct a response. Therefore, implementations SHOULD validate that an answer containing an overload report is a properly constructed response to a pending request prior to acting on the overload report.

A similar attack involves an otherwise authorized Diameter node that sends an inappropriate overload report. For example, a server for the realm "example.com" might send an overload report indicating that a competitor's realm "example.net" is overloaded. If other nodes act on the report, they may falsely believe that "example.net" is overloaded, effectively reducing that realm's capacity. Therefore, it's critical that nodes validate that an overload report received from a peer actually falls within that peer's responsibility before acting on the report or forwarding the report to other peers. For example, an overload report from a peer that applies to a realm not handled by that peer is suspect.

An attacker might use the information in an overload report to assist in certain attacks. For example, an attacker could use information about current overload conditions to time a DoS attack for maximum effect, or use subsequent overload reports as a feedback mechanism to learn the results of a previous or ongoing attack.

9.2. Denial of Service Attacks

Diameter overload reports can cause a node to cease sending some or all Diameter requests for an extended period. This makes them a tempting vector for DoS tacks. Furthermore, since Diameter is almost always used in support of other protocols, a DoS attack on Diameter is likely to impact those protocols as well. Therefore, Diameter nodes MUST NOT honor or forward overload reports from unauthorized or otherwise untrusted sources.

9.3. Non-Compliant Nodes

When a Diameter node sends an overload report, it cannot assume that all nodes will comply. A non-compliant node might continue to send requests with no reduction in load. Requirement 28 [RFC7068] indicates that the overload control solution cannot assume that all Diameter nodes in a network are necessarily trusted, and that malicious nodes not be allowed to take advantage of the overload control mechanism to get more than their fair share of service.

In the absence of an overload control mechanism, Diameter nodes need to implement strategies to protect themselves from floods of requests, and to make sure that a disproportionate load from one source does not prevent other sources from receiving service. For example, a Diameter server might reject a certain percentage of requests from sources that exceed certain limits. Overload control can be thought of as an optimization for such strategies, where downstream nodes never send the excess requests in the first place. However, the presence of an overload control mechanism does not remove the need for these other protection strategies.

9.4. End-to End-Security Issues

The lack of end-to-end security features makes it far more difficult to establish trust in overload reports that originate from non-adjacent nodes. Any agents in the message path may insert or modify overload reports. 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 Diameter overload control MUST give operators the ability to select which peers are trusted to deliver overload reports, and whether they are trusted to forward overload reports from non-adjacent nodes.

The lack of end-to-end confidentiality protection means that any Diameter agent in the path of an overload report can view the contents of that report. In addition to the requirement to select which peers are trusted to send overload reports, operators MUST be able to select which peers are authorized to receive reports. A node MUST not send an overload report to a peer not authorized to receive it. Furthermore, an agent MUST remove any overload reports that might have been inserted by other nodes before forwarding a Diameter message to a peer that is not authorized to receive overload reports.

At the time of this writing, the DIME working group is studying requirements for adding end-to-end security [I-D.ietf-dime-e2e-sec-req] features to Diameter. These features, when they become available, might make it easier to establish trust in non-adjacent nodes for overload control purposes. Readers should be reminded, however, that the overload control mechanism encourages Diameter agents to modify AVPs in, or insert additional AVPs into, 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 overload control will

require careful consideration, and are beyond the scope of this document.

10. Contributors

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

- Eric McMurry
- Hannes Tschofenig
- Ulrich Wiehe
- Jean-Jacques Trottin
- Maria Cruz Bartolome
- Martin Dolly
- Nirav Salot
- Susan Shishufeng

11. References

11.1. Normative References

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

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

[RFC5905] Mills, D., Martin, J., Burbank, J., and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, June 2010.

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

11.2. Informative References

[Cx] 3GPP, , "ETSI TS 129 229 V11.4.0", August 2013.

[I-D.ietf-dime-e2e-sec-req]
Tschofenig, H., Korhonen, J., Zorn, G., and K. Pillay, "Diameter AVP Level Security: Scenarios and Requirements", draft-ietf-dime-e2e-sec-req-00 (work in progress), September 2013.

[PCC] 3GPP, , "ETSI TS 123 203 V11.12.0", December 2013.

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

[RFC5729] Korhonen, J., Jones, M., Morand, L., and T. Tsou, "Clarifications on the Routing of Diameter Requests Based on the Username and the Realm", RFC 5729, December 2009.

[RFC7068] McMurry, E. and B. Campbell, "Diameter Overload Control Requirements", RFC 7068, November 2013.

[S13] 3GPP, , "ETSI TS 129 272 V11.9.0", December 2012.

Appendix A. Issues left for future specifications

The base solution for the overload control does not cover all possible use cases. A number of solution aspects were intentionally left for future specification and protocol work.

A.1. Additional traffic abatement algorithms

This specification describes only means for a simple loss based algorithm. Future algorithms can be added using the designed solution extension mechanism. The new algorithms need to be registered with IANA. See Sections 6.1 and 8 for the required IANA steps.

A.2. Agent Overload

This specification focuses on Diameter endpoint (server or client) overload. A separate extension will be required to outline the handling of the case of agent overload.

A.3. New Error Diagnostic AVP

The proposal was made to add a new Error Diagnostic AVP to supplement the error responces to be able to indicate that overload was the reason for the rejection of the message.

Appendix B. Deployment Considerations

Non supporting agents

Due to the way that realm-routed requests are handled in Diameter networks, with the server selection for the request done by an agent, it is recommended that deployments enable all agents that do server selection to support the DOIC solution prior to enabling the DOIC solution in the Diameter network.

Topology hiding interactions

There exist proxies that implement what is referred to as Topology Hiding. This can include cases where the agent modifies the Origin-Host in answer messages. The behavior of the DOIC solution is not well understood when this happens. As such, the DOIC solution does not address this scenario.

Appendix C. Requirements Conformance Analysis

This section contains the result of an analysis of the DOIC solutions conformance to the requirements defined in [RFC7068].

To be completed.

Appendix D. Considerations for Applications Integrating the DOIC Solution

This section outlines considerations to be taken into account when integrating the DOIC solution into Diameter applications.

D.1. Application Classification

The following is a classification of Diameter applications and request types. This discussion is meant to document factors that play into decisions made by the Diameter identity responsible for handling overload reports.

Section 8.1 of [RFC6733] defines two state machines that imply two types of applications, session-less and session-based applications. The primary difference between these types of applications is the lifetime of Session-Ids.

For session-based applications, the Session-Id is used to tie multiple requests into a single session.

The Credit-Control application defined in [RFC4006] is an example of a Diameter session-based application.

In session-less applications, the lifetime of the Session-Id is a single Diameter transaction, i.e. the session is implicitly terminated after a single Diameter transaction and a new Session-Id is generated for each Diameter request.

For the purposes of this discussion, session-less applications are further divided into two types of applications:

Stateless applications:

Requests within a stateless application have no relationship to each other. The 3GPP defined S13 application is an example of a stateless application [S13], where only a Diameter command is defined between a client and a server and no state is maintained between two consecutive transactions.

Pseudo-session applications:

Applications that do not rely on the Session-Id AVP for correlation of application messages related to the same session but use other session-related information in the Diameter requests for this purpose. The 3GPP defined Cx application [Cx] is an example of a pseudo-session application.

The handling of overload reports must take the type of application into consideration, as discussed in Appendix D.2.

D.2. Application Type Overload Implications

This section discusses considerations for mitigating overload reported by a Diameter entity. This discussion focuses on the type of application. Appendix D.3 discusses considerations for handling various request types when the target server is known to be in an overloaded state.

These discussions assume that the strategy for mitigating the reported overload is to reduce the overall workload sent to the overloaded entity. The concept of applying overload treatment to requests targeted for an overloaded Diameter entity is inherent to this discussion. The method used to reduce offered load is not specified here but could include routing requests to another Diameter entity known to be able to handle them, or it could mean rejecting certain requests. For a Diameter agent, rejecting requests will usually mean generating appropriate Diameter error responses. For a Diameter client, rejecting requests will depend upon the application. For example, it could mean giving an indication to the entity requesting the Diameter service that the network is busy and to try again later.

Stateless applications:

By definition there is no relationship between individual requests in a stateless application. As a result, when a request is sent or relayed to an overloaded Diameter entity - either a Diameter Server or a Diameter Agent - the sending or relaying entity can choose to apply the overload treatment to any request targeted for the overloaded entity.

Pseudo-session applications:

For pseudo-session applications, there is an implied ordering of requests. As a result, decisions about which requests towards an overloaded entity to reject could take the command code of the request into consideration. This generally means that transactions later in the sequence of transactions should be given more favorable treatment than messages earlier in the sequence. This is because more work has already been done by the Diameter network for those transactions that occur later in the sequence. Rejecting them could result in increasing the load on the network as the transactions earlier in the sequence might also need to be repeated.

Session-based applications:

Overload handling for session-based applications must take into consideration the work load associated with setting up and maintaining a session. As such, the entity sending requests towards an overloaded Diameter entity for a session-based application might tend to reject new session requests prior to rejecting intra-session requests. In addition, session ending requests might be given a lower probability of being rejected as rejecting session ending requests could result in session status being out of sync between the Diameter clients and servers. Application designers that would decide to reject mid-session requests will need to consider whether the rejection invalidates the session and any resulting session clean-up procedures.

D.3. Request Transaction Classification

Independent Request:

An independent request is not correlated to any other requests and, as such, the lifetime of the session-id is constrained to an individual transaction.

Session-Initiating Request:

A session-initiating request is the initial message that establishes a Diameter session. The ACR message defined in [RFC6733] is an example of a session-initiating request.

Correlated Session-Initiating Request:

There are cases when multiple session-initiated requests must be correlated and managed by the same Diameter server. It is notably the case in the 3GPP PCC architecture [PCC], where multiple apparently independent Diameter application sessions are actually correlated and must be handled by the same Diameter server.

Intra-Session Request:

An intra session request is a request that uses the same Session-Id than the one used in a previous request. An intra session request generally needs to be delivered to the server that handled the session creating request for the session. The STR message defined in [RFC6733] is an example of an intra-session requests.

Pseudo-Session Requests:

Pseudo-session requests are independent requests and do not use the same Session-Id but are correlated by other session-related information contained in the request. There exists Diameter applications that define an expected ordering of transactions. This sequencing of independent transactions results in a pseudo session. The AIR, MAR and SAR requests in the 3GPP defined Cx [Cx] application are examples of pseudo-session requests.

D.4. Request Type Overload Implications

The request classes identified in Appendix D.3 have implications on decisions about which requests should be throttled first. The following list of request treatment regarding throttling is provided as guidelines for application designers when implementing the Diameter overload control mechanism described in this document. The exact behavior regarding throttling is a matter of local policy, unless specifically defined for the application.

Independent requests:

Independent requests can generally be given equal treatment when making throttling decisions, unless otherwise indicated by application requirements or local policy.

Session-initiating requests:

Session-initiating requests often represent more work than independent or intra-session requests. Moreover, session-initiating requests are typically followed by other session-related requests. Since the main objective of the overload control is to reduce the total number of requests sent to the overloaded entity, throttling decisions might favor allowing intra-session requests over session-initiating requests. In the absence of local policies or application specific requirements to the contrary, Individual session-initiating requests can be given equal treatment when making throttling decisions.

Correlated session-initiating requests:

A Request that results in a new binding, where the binding is used for routing of subsequent session-initiating requests to the same server, represents more work load than other requests. As such, these requests might be throttled more frequently than other request types.

Pseudo-session requests:

Throttling decisions for pseudo-session requests can take into consideration where individual requests fit into the overall sequence of requests within the pseudo session. Requests that are earlier in the sequence might be throttled more aggressively than requests that occur later in the sequence.

Intra-session requests:

There are two types of intra-sessions requests, requests that terminate a session and the remainder of intra-session requests. Implementors and operators may choose to throttle session-terminating requests less aggressively in order to gracefully terminate sessions, allow clean-up of the related resources (e.g. session state) and avoid the need for additional intra-session requests. Favoring session-termination requests may reduce the session management impact on the overloaded entity. The default handling of other intra-session requests might be to treat them equally when making throttling decisions. There might also be application level considerations whether some request types are favored over others.

Authors' Addresses

Jouni Korhonen (editor)
Broadcom
Porkkalankatu 24
Helsinki FIN-00180
Finland

Email: jouni.nospam@gmail.com

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

Email: srdonovan@usdonovans.com

Ben Campbell
Oracle
7460 Warren Parkway
Frisco, Texas 75034
United States

Email: ben@nostrum.com

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

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