Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: January 7, 2016

B. Campbell
S. Donovan, Ed.
Oracle
JJ. Trottin
Alcatel-Lucent
July 6, 2015

Diameter Load Information Conveyance
draft-ietf-dime-load-00

Abstract

This document defines a mechanism for sharing of Diameter load information. RFC 7068 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) 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 January 7, 2016.

Copyright Notice

Table of Contents

1. Introduction

[RFC7068] describes requirements for Overload Control in Diameter [RFC6733]. At the time of this writing, the DIME working group is working on the Diameter Overload Information Conveyance (DOIC)

mechanism [I-D.ietf-dime-ovli] . 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 RFC 7068 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 defer 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

Load

The relative capacity of a Diameter node. A low value indicates that the Diameter node is under utilized. A high value indicated 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.

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 have 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) [I-D.ietf-dime-ovli] 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 [I-D.ietf-dime-ovli]. 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.

> Editor's Note: One area that requires thought is how load information is used, if at all, in the presence of an overload report from the same Diameter node. It might be that the load information from that Diameter node is ignored for the duration of the time that the overload report is in effect. It might also be possible that the load information can aid in the diverting of non-abated requests targeted for the overloaded Diameter node.

5. Solution Overview

The mechanism defined here for the conveyance of load information is similar in some says 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 extension then the load information is ignored.

The second big difference between DOIC and Load is that DOIC information is sent end-to-end. 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, the load information is targeted for the peer of the sender of the Load AVPs.

> Editor's note: It is still being discussed whether a second type of Load report can be included that is the load of the end-point and is carried end-to-end.

The Load report applies to an individual node. The Diameter identity of the node to which it applies is included in the Load report.

> Note, this is necessary so that supporting nodes can determine if the load report came from a peer node. If the identity is for a non peer node than the peer load report can be ignored.

In addition to the identity of the node to which the report applies, 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 method for calculating the load value included in the load report is 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 value. 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:

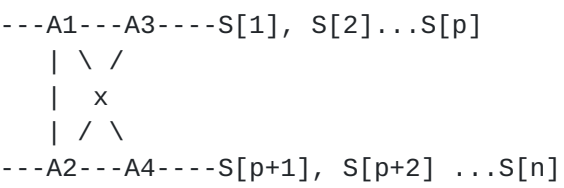

Figure 1: Example Diameter Network

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

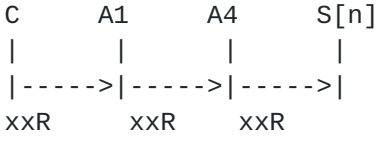

Figure 2: Request Message Path

When sending the answer message, a sending node that supports the Diameter Load mechanism includes it's own load information in the answer message:

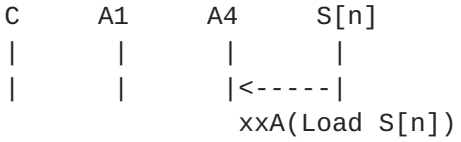

Figure 3: Answer Message from S[n]

If Agent A4 supports the Load mechanism then it will verify that the load information received was from its peer. This is achieved by matching the identity included in the load information with the identity of the peer node from which the answer message was received.

If the load information does not match then the agent will strip the load AVPs from the message.

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

In all cases, A4 strips the load AVPs from the message.

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

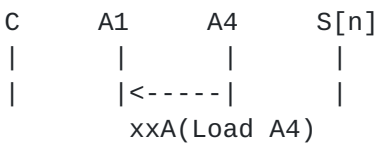

Figure 4: Answer Message from A4

A1 follows the same procedures as A4, resulting in the following message sent to C:

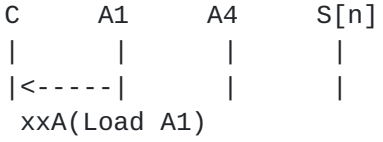

Figure 5: Answer Message from A1

C follows the same procedure 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.

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

Editor's note: The above needs to be updated if there is agreement to support end-point Load reports.

6. Solution Procedures

6.1. Reporting Node Behavior

6.1.1. Endpoint Reporting Node Behavior

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

This document makes no requests of IANA.

10. References

10.1. Normative References

[I-D.ietf-dime-ovli] Korhonen, J., Donovan, S., Campbell, B., and L. Morand, "Diameter Overload Indication Conveyance", draft-ietf-dime-ovli-03 (work in progress), July 2014.

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

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

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

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, February 2000.

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.

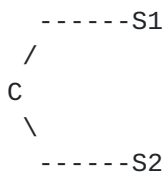

Figure 6: Basic Client Server Scenario

Open Issue: Will a Diameter node include potential peers that it is not currently connected to as part of the candidate set? It is unlikely the client would have load information from peers that it is not currently connected to.

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.

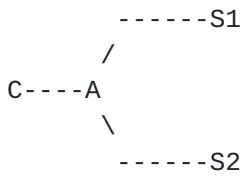

Figure 7: Simple Agent Scenario

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 [I-D.ietf-dime-ovli], 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.

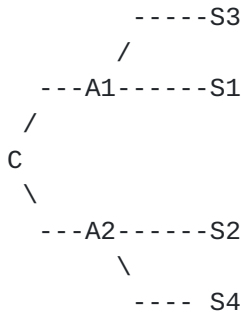

Figure 8: Multiple Agents and Servers

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.

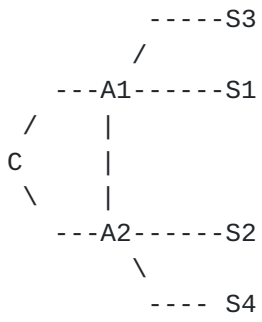

Figure 9: Linked Agents

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.

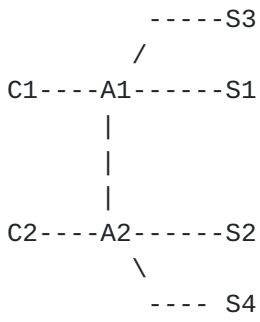

Figure 10: Linked Agents

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.

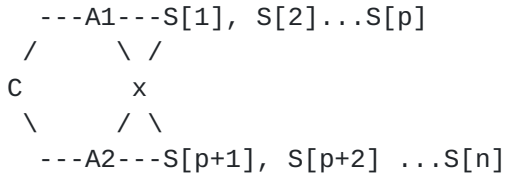

Figure 11: Shared Server Pools

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.

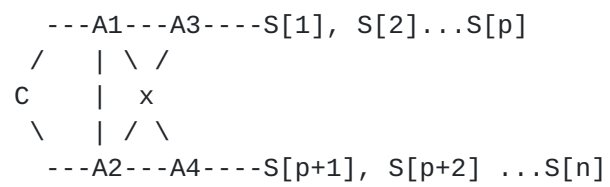

Figure 12: Agent Chains

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.

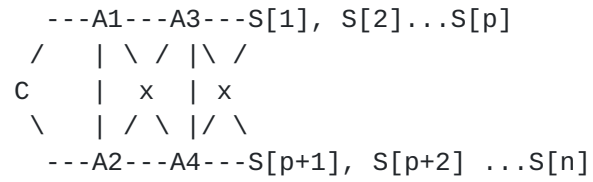

Figure 13: Full Mesh

A.8. Partitions

A Diameter network with multiple is said to be "partitioned" when only a subset of available servers can server 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. It should be determined if this use case is in the scope of the load control mechanism.

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. It should be determined if this use case is in the scope of the load control mechanism.

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
Alcatel-Lucent
Route de Villejust
91620 Nozay
France

Email: jean-jacques.trottin@alcatel-lucent.com