                     Diameter Agent Overload
                 draft-ietf-dime-agent-overload-02.txt

Abstract

   This specification documents an extension to the Diameter Overload
   Indication Conveyance (DOIC) base solution. The extension addresses
   the handling of occurrences of overload of a Diameter agent, or more
   generally, a Diameter peer.

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 March 3, 2016.

Table of Contents

1. Introduction

   This document defines the behavior of Diameter nodes when Diameter
   agents enter an overload condition and send an overload report
   requesting a reduction of traffic.

The base Diameter overload specification [I-D.ietf-dime-ovli] addresses the handling of overload when a Diameter endpoint (a Diameter Client or Diameter Server as defined in [RFC6733]) becomes overloaded.

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

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

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

The handling of endpoint overload and agent overload is very similar. The primary differences are the following:

o Endpoint overload is handled as close to the originator of the traffic as possible.

o Agent overload is handled by the previous hop Diameter Node.

o Endpoint overload mitigation deals with traffic targeted for a single Diameter application. As such, it is assumed that an overload report impacts just the application implied by the message carrying the overload report.

o Agent overload deals with all traffic targeted for an agent, independent of the application. As such, a single agent overload report can impact multiple applications.

Editor's Note: Open Issue - Does a peer report apply to the implicitly communicated application-id in the same way as host and

realm reports do or does it apply to all applications handled by the
peer? Do we need the ability for to support both cases?

Open Issue - To support the ability of an agent to select a different
abatement algorithm than endpoints, we probably need to extend the
OC-Supported-Features AVP to include an OC-Abatement-Algorithm AVP.
This is currently shown to be in the OC-OLR AVP but needs to be moved
as this information is needed prior to receiving the OC-OLR. It
probably needs to be changed to OC-Peer-Abatement-Algorithm.

[2](). Terminology and Abbreviations

Editors note - These definitions need to be made consistent with the
base Diameter overload specification defined in [I-D.ietf-dime-ovli].

Diameter Node

   A [RFC6733]() Diameter Client, an [RFC6733]() Diameter Server, and [RFC6733]()
   Diameter Agent.

Diameter Endpoint

   An [RFC6733]() Diameter Client and [RFC6733]() Diameter Server.

Reporting Node

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

Reacting Node

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

DIOC Node

   A Diameter Node that supports the DOIC solution defined in
   [I-D.ietf-dime-ovli].

[3](). Peer Report Use Cases

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

There are two primary classes of use cases, those involving the
overload of agents and those involving overload of Diameter endpoints
(Diameter Clients and Diameter Servers).

3.1. **Diameter Agent Overload Use Cases**

   The agent overload extension must support the following use cases.

3.1.1. **Single Agent**

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

```
          +-+     +-+     +-+
          |c|----|a|----|s|
          +-+     +-+     +-+
```
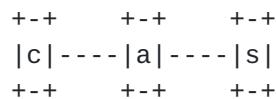
                           Figure 1

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

```
                          +-+
                        --|s|
          +-+     +-+ /   +-+
          |c|----|a|-    ...
          +-+     +-+ \   +-+
                        --|s|
                          +-+
```
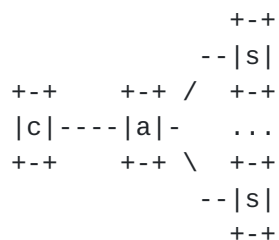
                           Figure 2

   In both of these cases, the occurrence of overload in the single
   agent must by handled by the client in a similar fashion as if the
   client were handling the overload of a directly connected server.
   When the agent becomes overloaded it will insert an overload report
   in answer messages flowing to the client. This overload report will
   contain a requested reduction in the amount of traffic sent to the
   agent. The client will apply overload abatement behavior as defined
   in the base Diameter overload specification [I-D.ietf-dime-ovli] or
   the extension draft that defines the indicated overload abatement
   algorithm. This will result in the throtting of the abated traffic
   that would have been sent to the agent, as there is no alternative
   route, with the appropriate indication given to the service request
   that resulted in the need for the Diameter transaction.

[3.1.2](#). **Redundant Agents**

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

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

```
              +--+    +-+
             --|a1|---|s|
        +-+ /   +--+\ /+-+
        |c|-         x
        +-+ .   +--+/ \+-+
             ..|a2|---|s|
               +--+    +-+
```
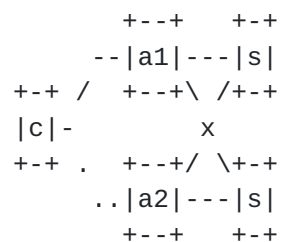

                         Figure 3

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

```
              +--+    +-+
             --|a1|---|s|
        +-+ /   +--+\ /+-+
        |c|-         x
        +-+ \   +--+/ \+-+
             --|a2|---|s|
               +--+    +-+
```
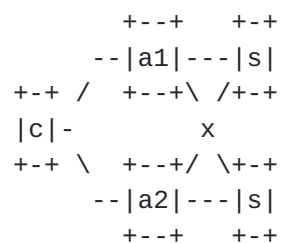

                         Figure 4

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

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

In the case where both agents are reporting overload, the client may
need to start decreasing the total traffic sent to the agents. This
would be done in a similar fashion as discussed in section 3.1. The
amount of traffic depends on the combined reduction requested by the
two agents.

3.1.3. Agent Chains

There are also deployment scenarios where there can be multiple
Diameter Agents between Diameter Clients and Diameter Servers.
Examples of this type of deployment include when there are edge
agents between Diameter networks. Another example of this type of
deployment is when there are multiple sets of servers, each
supporting a subset of the Diameter traffic.

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

```
            +---+      +---+   +-+
          --|a11|-----|a21|---|s|
    +-+ /   +---+ \ / +---+\ /+-+
    |c|-           X        X
    +-+ \   +---+ / \ +---+/ \+-+
          --|a12|-----|a22|---|s|
            +---+      +---+   +-+
```
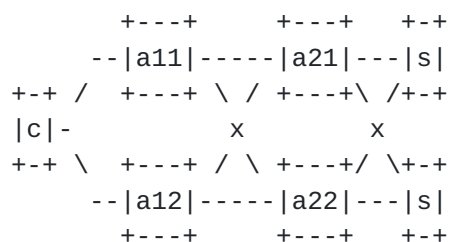
Figure 5

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

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

Editor's note: Probably need to elaborate the reasoning behind the
need for the agent overload report being handled by the previous hop
agent.

The handling of peer overload reports is similar to that discussed in
section 2.2. If the overload can be addressed using diversion then
this approach should be taken.

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

3.2. Diameter Endpoint Use Cases

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

3.2.1. Hop-by-hop Abatement Algorithms

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

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

4. Interaction Between Host/Realm and Peer Overload Reports

It is possible that both an agent and a server in the path of a
transaction are overloaded at the same time. When this occurs,
Diameter entities will need to handle both overload reports. In this
scenario the reacting node should first handle the throttling of the
overloaded host or realm. Any messages that survive throttling due
to host or realm reports should then go through abatement for the
peer overload report.

5. Peer Report Behavior

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

5.1. Capability Announcement

Editor's Note: Issue - how does an agent indicate the selected
abatement algorithm? It cannot use the OC-Feature-Vector in the OC-

Supported-Features AVP as that applies to host and realm report
types. A new AVP in the OC-Supported-Features AVP has been added.

5.1.1. Reacting Node Behavior

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

> Note: The sender of a request can be a Diameter Client or Diameter
> Server that originates the Diamter request or a Diameter Agent
> that relays the request.

Support for the peer report feature does not impact the logic for
setting of other feature bits in the OC-Feature-Vector AVP.

When sending a request a DOIC node that supports the Peer Report
feature MUST include an OC-SourceID AVP in the OC-Supported-Features
AVP with its own DiameterID.

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

5.1.2. Reporting Node Behavior

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

> Note" The transaction state is used when the DOIC node is acting
> as a peer-report reporting node and needs to insert OC-Supported-
> Feature AVP indicating support for the OLR_PEER_REPORT feature and
> OC-OLR reports of type PEER_REPORT into answer messages. These
> AVP OLR are only included in answer messages being sent to peers
> that support the OLR_PEER_REPORT feature.

The following are indications that the peer does not support the
OLR_PEER_REPORT feature:

> The request does not contain an OC-Supported-Features AVP.

> The received request contains an OC-Supported-Features AVP with no
> OC-Feature-Vector.

The received request contains an OC-Supported-Features AVP with a
OC-Feature-Vector with the OLR_PEER_REPORT feature bit cleared.

The received request contains an OC-Supported-Features AVP with a
OC-Feature-Vector with the OLR_PEER_REPORT feature bit set but
with an OC-SourceID AVP with a DiameterID that does not match the
DiameterID of the peer from which the request was received.

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

When receiving a request a DOIC node that supports the Peer Report
feature MUST remove any received OC-SourceID AVP from the OC-
Supported-Features AVP. This is done to prevent the OC-SourceID AVP
from being included in a relayed message through a node that supports
the Peer Report feature.

Note: If the DOIC node relays the message then it will insert an
OC-SourceID AVP with its own DiameterID in the OC-Supported-
Features AVP in the relayed message.

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

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

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

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

If the peer supports the OLR_PEER_REPORT feature then the reporting
node MUST insert the OC-Peer-Algo AVP in the OC-Supported-Features
AVP. The OC-Peer-Algo AVP MUST indicate the overload abatement
algorithm that the reporting node wants reacting nodes to use should
the reporting node send a peer overload report as a result of
becoming overloaded.

5.2. Peer Report Overload Report Handling

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

5.2.1. Overload Control State

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

5.2.1.1. Reporting Node Peer Report OCS

   A DOIC Node that supports the Peer Report feature SHOULD maintain
   Reporting Node Peer Report OCS. This is used to record overload
   events and build overload reports at the reporting node.

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

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

   The Reporting Node Peer Report OCS entry MAY include the following
   information (the actual information stored is an implementation
   decision):

   o Sequence number

   o Validity Duration

   o Expiration Time

   o Abatement Algorithm

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

5.2.1.2. Reacting Node Peer Report OCS

   A DOIC node that supports the Peer Report feature SHOULD maintain
   Reacting Node Peer Report OCS for each peer with which it
   communicates. This is used to record overload reports received from
   peer nodes.

A Reacting Node Peer Report OCS entry is identified by the DiameterID
of the peer as communicated during the RFC6733 defined Capability
Exchange procedure

The Reacting Node Peer Report OCS entry MAY include the following
information (the actual information stored is an implementation
decision):

o Sequence number

o Expiration Time

o Abatement Algorithm

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

5.2.2. Reporting Node Maintenance of Peer Report OCS

A reporting node SHOULD create a new Reporting Node Peer Report OCS
entry Section 5.2.1.1 in an overload condition and sending a peer
overload report to a peer for the first time.

> If the reporting node knows that there are no reacting nodes
> supporting the Peer Report feature then the reporting node can
> choose to not create OCS entries.

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

5.2.3. Reacting Node Maintenance of Peer Report OCS

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

If the DiameterID in the SourceID contained in the OLR matches the
DiameterID of the peer from which the request was received then the
report was received from a Diameter peer.

If a reacting node receives an OC-OLR AVP of type peer and the OC-
SourceID does not match the ID of the Diameter peer from which the
request was received then the reacting node MUST strip the OC-OLR AVP
from the message and not use it to update reacting node peer report
OCS entries.

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

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

For a peer report-type this means the DiameterID received in the
SourceID AVP matches the DiameterID of an existing peer report OLR.

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.

   Editor's note: The above four paragraphs are copied form the DOIC
   specification. Is it possible to include this behavior by
   reference or do we need to include all of these statements in this
   specification as well.

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

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

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

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

5.2.4. Peer Report Reporting Node Behavior

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

   The reporting node determines if a peer supports the peer report
   feature based on the indication recorded in the reporting nodes
   transaction state.

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

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

5.2.5. Peer Report Reacting Node Behavior

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

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

If a host-routed request, as defined in [I-D.ietf-dime-ovli], is
selected for abatement and the request must be routed to the DOIC
node that generated the peer overload report -- meaning that the
request is a host-routed request as defined in the DOIC specification
-- then the reacting node MUST throttle the request.

   This would result from an overloaded Diameter endpoint (Diameter
   Server or Diameter Client) sending a peer overload report and the
   request contains a Destination-Host AVP with a DiameterID that

matches the DiameterID in the SourceID AVP received in the peer
overload report.

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

If the abatement treatment results in throttling of the request and
if the reacting node is an agent then the agent MUST send an
appropriate error as defined in [I-D.ietf-dime-ovli].

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.

6. Peer Report AVPs

6.1. OC-Supported-Features AVP

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

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

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

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

6.1.1. OC-Feature-Vector

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

OLR_PEER_REPORT (0x0000000000000010)

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

6.1.2. OC-Peer-Algo

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

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

   Editor's node: This is to avoid the need for an additional IANA
   registry.

6.2. OC-OLR AVP

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

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

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

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

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

6.2.1. OC-Report-Type AVP

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

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

6.3. OC-SourceID

The SourceID AVP (AVP code TBD) is of type DiameterIdentity and is
inserted by the DOIC node that either indicates support for this
feature (in the OC-Supported-Features AVP) or that generates an OC-
OLR AVP with a report type of peer.

It contains the Diameter Identity of the inserting node. This is
used by other DOIC nodes to determine if the a peer indicated support
this feature or inserted the peer report.

6.4. Attribute Value Pair flag rules

```
                                             +---------+
                                             |AVP flag |
                                             |rules    |
                                             +----+----+
                           AVP   Section     |    |MUST|
    Attribute Name         Code  Defined Value Type   |MUST| NOT|
   +-------------------------------------------------+----+----+
   |OC-SourceID            TBD1   x.x    Unsigned64   |    | V  |
   |OC-Peer-Algo           TBD1   x.x    Unsigned64   |    | V  |
   +-------------------------------------------------+----+----+
```

7. IANA Considerations

Editors note: This section will be completed once the base overload
document has finished the definition of extension IANA requirements.

8. Security Considerations

Agent overload is an extension to the based Diameter overload
mechanism. As such, all of the security considerations outlined in
[I-D.ietf-dime-ovli] apply to the agent overload scenarios.

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

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

9. Acknowledgements

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

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

10. Normative References

[I-D.ietf-dime-doic-rate-control]
          Donovan, S. and E. Noel, "Diameter Overload Rate Control",
          draft-ietf-dime-doic-rate-control-01 (work in progress),
          March 2015.

[I-D.ietf-dime-ovli]
          Korhonen, J., Donovan, S., Campbell, B., and L. Morand,
          "Diameter Overload Indication Conveyance", draft-ietf-
          dime-ovli-08 (work in progress), February 2015.

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

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

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

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

Author's Address

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

   Email: srdonovan@usdonovans.com