SVI Diameter Mediation User Guide

Document SQ-UG-030

Version 1.7

Change History

Version	Change Made	Author	Authorised	Date
1.0	Initial Release	DR	AC	25.04.17
1.1	Added Store-AVP	JF	AC	20.08.17
1.2	Added additional examples	JF	AC	12.12.17
1.3	Updated to include session variables	SW	AC	02.05.18
1.4	Applied latest corporate formatting	SS	SS	19.02.20
1.5	Updates to triggers	SA	SS	18.06.20
1.6	NotEqual operation corrections	SA	SS	21.10.20
1.7	Added Route, updated 3.6	SA	SS	18.01.21

Contents

Introduction

The SVI-DSC provides a mediation function that allows for the manipulation and steering of Diameter messages solving interoperabiity issues, providing access control, network topolgy hiding and advanced routing among other functions.

Mediation is implemented by the creation of Rules that allow for the inspection of the Diameter message against a mach criteria.

Mediation is defined by XML file(s); these files are used by XML resources within SVI. These XML resources must be attched to Diameter Remote Peers that originate the Request (regardless of trigger type).

The match criteria can be made on the following

- Diameter Header contents
- AVP contents
- If the AVP is present or not

The contents of the AVP can be matched against a number of given criteria

- Exact match
- Ranges
- lists
- Operations like smaller,larger, not equal
- Regular expression matches.

Multiple matches can be defined and formed into a logical condition using AND and OR allowing for complex matches to be made to determine if an action should happen.

If the match criteria is met then the rules actions are undertaken on the Diameter message, this includes

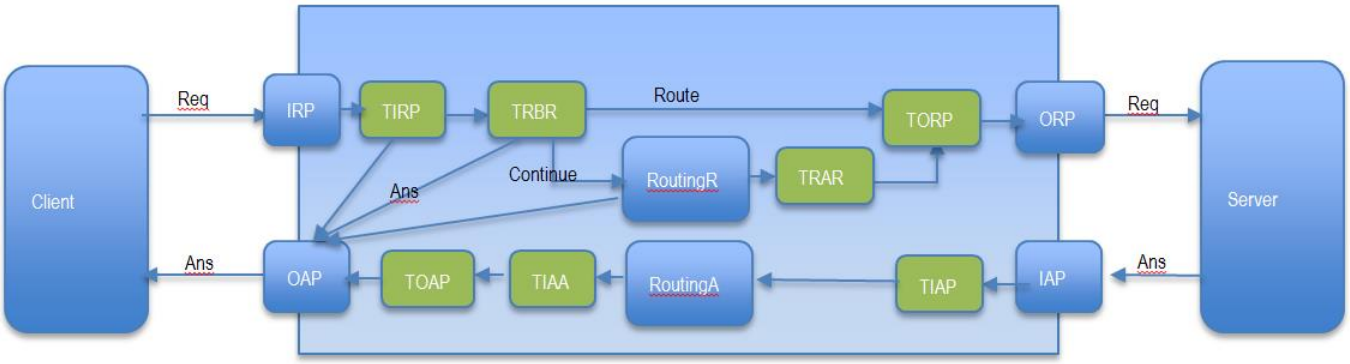
- Modifying Diameter Header
- Modifying,Removing or Adding AVPS
- Rejecting or Ignoring the transaction
- Answering the Transaction

It is also possible to store the contents of the Diameter message to modify or Add other AVPs from the contents of a particular AVP. These stored variables can then be used at a future date within a transaciton or session allowing for modification of a Diameter messages from these earlier stored variables.

Rules can also be linked to apply multiple rules to a single Diameter message.

1 Trigger Points

The SVI-DSC incorporates a number of trigger points that the mediation can be applied to. The following diagram shows for a transaction flow where these trigger points are



Component	Name	Type	Description
IRP	Incoming Request Peer	Function	Point on reception of request at peer after general message compliance
ORP	Outing Request Peer	Function	Entry point into outgoing request transmission queue
IAP	Incoming Answer Peer	Function	Entry point receiving incoming answer
OAP	Outgoing Answer Peer	Function	Entry point into outgoing answer request transmission queue
RoutingR	Routing Receive	Function	Apply routing table logic
RoutingA	Routing Answer	Function	Exit point of answer routing logic on selection of incoming answer
TIRP	Trigger Incoming Request Peer	Trigger	Trigger after IRP
TOAP	Trigger Outgoing Answer Peer	Trigger	Trigger before OAP
TRBR	Trigger Request Before Routing	Trigger	Trigger before RoutingA
TRAR	Trigger Request After Routing	Trigger	Trigger After RoutingA
TORP	Trigger Outgoing Request Peer	Trigger	Trigger Before ORP
TIAP	Trigger Incoming Answer Peer	Trigger	Trigger After RoutingA
TIAA	Trigger Incoming Answer Accepted	Trigger	Trigger point for when transaction answer has been accepted from routing

2 Mediation Overview

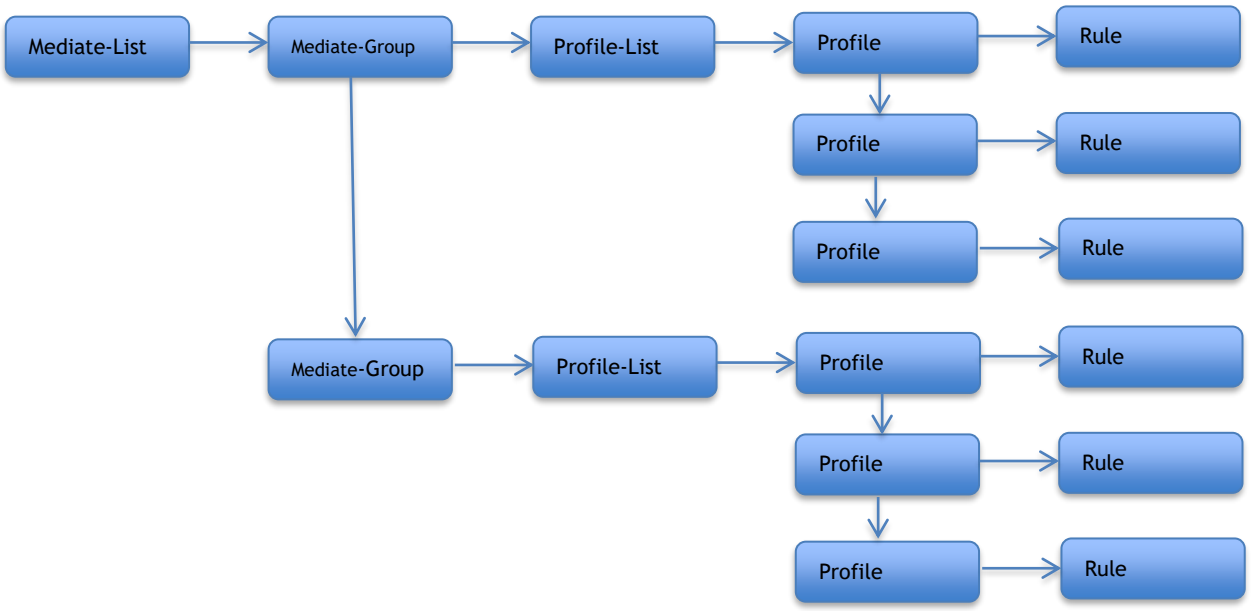
The mediation rules are modelled using an XML format. There are 5 main elements used to Mediate-List, Media-Group, Profile-List, Profile and Rule

The Mediate-List is the top XML element. Nested below the Mediate-List element is the Media-Group element. Multiple Media-Group elements cans be nested below the Mediate-List. To decide which Mediate-Group to run the required header contents for this Mediate-Group can be attached.

Nested below the Mediate-Group is the Profile-List. This contains a list of profile elements. The profile elements controls which order the Rules are run in. The top profile in the profile list is run first. Depending on the result of the Rule (Match or No-Match) the profile can either jump to another rule or exit from the mediation session.

The rule element contains a match element that if matched will run its Action elements and return Match back to the Profile. If no match is found in this rule then the rule will return No-Match

The diagram below shows a diagrammatic representation of the above.



3 Mediation Elements

The following describes the format of the Mediation XML Elements. The tables provide an indication of the XML schema.

The type column describes if the Elements attribute is a nested element (Element) or an XML-Attribute (Attribute)

```
<Element attribute="contents">
```

The type also defines the type of contents on an attribute. This is either a String or an ENUM. If it is described as an ENUM then the possible values of the contents are given in the Description Column. A string can contain either ASCII text or a single Integer or an Integer Range. The Integer range has a format of "1,2-10,11".

For some attributes (Header and AVP contents) the ENUM value is taken from the Diameter Dictionary as loaded into the SVI-DSC. For more details on the Diameter Dictionary see the SVI-DSC user guide.

Comments can be also added into the XML by the following syntax

```
<!-- comment entered here-->
```

3.1 Mediate-List

The Mediate-List element is the top element of the Mediation XML file.

The Mediate-List element syntax is <Mediate-List>

Attribute	Type	Options	Description
Description	Attribute String		Contains unique description of the Mediate-List entry
Mediate-Group	Element	Repeatable	Mediate-Group

3.1.1 Example

```
<Mediate-List>

</Mediate-List>
```

3.2 Mediate-Group

The Mediate-Group element contains a group of rules and the order in which these rules are executed. Multiple Mediate-Group elements can be defined in a Mediate-List element. If multiple Mediate-Group elements are present, which Mediate-Group is to be used is then identified by the TP (Trigger Point) attribute and the Matches element. Only one Mediate-Group will be run in a trigger session.

The Mediate-Group element syntax is <Mediate-Group>

Attribute	Type	Options	Description
TP	Attribute ENUM		Contains the trigger point that this Mediate-Group will be actioned on. Values: _TP_IRP : Incoming Request On Peer _TP_ORP : Outgoing Request on Peer _TP_IAP : Incoming Answer on Peer _TP_OAP : Outgoing Answer on Peer _TP_RBR : Request before Routing _TP_RAR : Request after Routing
Matches			Contains matching information for this Mediate- Group to be selected from Mediate-List See Matches
Profile-List	Element	Optional	Contains a list of profiles identifying in which order the rules are executed See Profile-List
Rule-List	Element	Mandatory	Contains a list of rules which are executed depending on the profiles. See Rule-List

3.2.1 Examples

The following example shows the configuration of a single Mediate-Group

```
<Mediate-List>
  <Mediate-Group>

  </Mediate-Group>
</Mediate-List>
```

This next example shows Media groups for an incoming request and outgoing answer on a peer.

```
  <Mediate-List>
    <!-- Run this Mediate-Group for trigger point _TP_IRP for interface 3GPP S6a/S6d-->
    <Mediate-Group TP="_TP_IRP">
      <Matches>
        <Message-Match>
          <Header appid="3GPP S6a/S6d"/>
        </Message-Match>
```

```
    </Matches>
  </Mediate-Group>

  <!-- Run this Mediate-Group for trigger point _TP_OAP for interface 3GPP S6a/S6d-->
  <Mediate-Group TP="_TP_OAP">
    <Matches>
      <Message-Match>
        <Header appid="3GPP S6a/S6d"/>
      </Message-Match>
    </Matches>
  </Mediate-Group>

</Mediate-List>
```

3.3 Profile-List

The Profile-List element contains a list of profile elements that determine in which order the rule elements should be executed. The first profile in the list will be executed first. The profile then identifies, depending on the result back from the profile's attached rule, if another profile is to be executed or the mediation session has ended.
If the profile-list is not configured then the rule elements are run in list order.

The Profile-List element syntax is <Profile-List>

Attribute	Type	Options	Description
Profile	Element	Repeatable	See Profile

3.3.1 Profile

The Profile element indicates which rule should be run and depending on the rule element's result the action to be undertaken. This can either be run a different rule or exit from the mediation session with a given result

The Profile element syntax is <Profile>

Attribute	Type	Options	Description
Name	Attribute String	Mandatory	This contains the name of the profile
Rule	Attribute String	Mandatory	This contains the name of the rule attached to this profile which is included in the parent Mediate-Group rule-list element.
onMatchExit	Attribute Boolean	Optional	If the result of the rule is Match and this attribute is set to True then the trigger session will be ended. Values: True False
OnNoMatchExit	Attribute ENUM	Optional	If the result of the rule is NoMatch and this attribute is set to True then the trigger session will be ended. Values: True False
MatchRule	Attribute String	Optional	If the result of the rule is Match and this attribute contains the name of another profile in the list, this new profile will be executed.
NoMatchRule	Attribute String	Optional	If the result of the rule is NoMatch and this attribute contains the name of another profile in the list, this new profile will be executed.

3.3.2 Example

The following example shows a simple Profile list that will execute the first profile,"profile1", which will run rule1. If rule1 returns match then exit the mediation session. If the first rule does not match run "profile 2" which will run "rule2" and exit the mediation session

```
<Mediate-List>
  <!-- Run this Mediate-Group if this is a Credit-Control Request-->
  <Mediate-Group code="Credit-Control" flags.req="True">
    <Profile-List>
      <Profile name="Profile1" rule="rule1" OnMatchExit="True" NoMatchRule="Profile2"/>
      <Profile name="Profile2" rule="rule2" OnMatchExit="True" NoNoMatchExit="True"/>
    </Profile-List>
```

```
      <Rule-List>
        <Rule name="rule1">

        </Rule>
        <Rule name="rule2">

        </Rule>
      </Rule-List>
    </Mediate-Group>
  </Mediate-List>
```

3.4 Rule-List

The Rule-List contains a list of rule elements

The Rule-List element syntax is <Rule-List>

Attribute	Type	Options	Description
Rule	Element	Repeatable	See Rule

3.4.1 Rule

The Rule element contains a Matches element that if the result is positive one of the Action elements is executed. Only on action element can be defined. If no matches' element is present then the action is executed. If no action element is present then the mediation session will continue

The Rule element syntax is <Rule>

Attribute	Type	Options	Description
Matches	Element	Optional	This contains a list of matches that must be True for the rule to Match. If this element is not set then the rule's result will be Match. Matches
Action-Modify	Element	Optional	This action will modify the message and continue the mediation session See Action-Modify
Action-Reject	Element	Optional	This action will reject a request message and exit the mediation session See Action-Reject
Action-Ignore	Element	Optional	This action will ignore the message and exit the mediation message See Action-Ignore
Action-Answer	Element	Optional	This action will create an answer message to be returned back to the originating request. The mediation session will be terminated See Action-Answer
Action-Route	Element	Optional	This action will modify the incoming request and allow for the routing table entry to be specified for the outgoing route See Action-Route

3.5 Matches

The Matches element contains a list of match elements. The exp (expression) attribute can be used to apply a logical expression to the matches list enabling complex inspection of the incoming Diameter message. For the expression functionality to work the match element with in the list must have its name defined. If the exp attribute is not set he match elements are executed in order until a match on a particular entry is identified at which point the match actions are executed.

The Matches element syntax is <Matches>

Attribute	Type	Options	Description
cond	Attribute String	Optional	This contains a bracketed AND/OR logical condition allowing for named Message-Match and Peer-Match elements to be formed into a logical expression. (matcha AND matchb) OR matchc
Message-Match	Element	Optional	Contains a list of Message-Match elements. See Message-Match
Peer-Match	Element	Optional	Contains incoming and outgoing peer information See Peer-Match
Carrier-Match	Element	Optional	Contains a match against an IMSI or MCC/MNC string against the Carrier List database to determine if access to/from this carrier is valid See Carrier-Match

3.5.1 Message-Match

The Message-Match element contains a description of a Diameter message to be compared against the incoming Diameter message.

The Message-Match element syntax is `<Message-Match>`

Attribute	Type	Options	Description
Name	Attribute String		If a logical expression is defined in the Matches parent, the name field must be set to the same name in the Matches exp attribute
Header	Element	Repeatable Optional	Header-Match
AVP	Element	Repeatable Optional	AVP-Match

The Header element can be repeated within the Message-Match element. For the Match to be True one of the Header-Elements must match.

Header-Matches and AVP-Matches cannot both be added at this level. Either Header-Matches are defined at this level with corresponding AVP-Matches for this message, or AVP-Matches are defined for any message.

3.5.2 Header-Match

The Header-Match element provides a description of the Diameter header that needs to be matched.

The Message-Match element syntax is `<Header>`

Attribute	Type	Options	Description
Appid	Attribute		Range for matching Application Id. The format can either be integer or text if application Id is specified in the Dictionary
Length	Attribute		Integer value
Code	Attribute		Range for matching codes The format can either be integer or text id the code is specified in the dictionary
Flags.e	Attribute		Boolean value for the error bit in the header flags. False : 0 True : 1
Flags.t	Attribute		Boolean value for the retransmitted bit in the header flags. False : 0 True : 1
Flags.r	Attribute		Boolean value for the request bit in the header flags. False : 0 True : 1
Version	Attribute		Range of integer values
HbH	Attribute		Integer Value
E2E	Attribute		Integer Value
AVP	Element	Repeatable Optional	Used to identify the contents of a Grouped AVP See AVP-Match

If AVP elements are added to the Header-Match then for the match to be true all of the defined AVP matches must be True.

3.5.2.1 Example

The following shows some examples for different header matches

```
<!--Match on message header with dictionary name for a request-->
<Header code="Credit-Control" flags.r="True"/>
<!--Match on message header with code as an integer-->
```

```
<Header code="272" flags.r="True"/>
<!--Match on message header with range as dictionary and integer-->
<Header code="Abort-Session,Session-Terminated,Credit-Control,258" flags.r="True"/>
<!--Match on application id from dictionary for all answers-->
<Header appid="Sh" flags.r="True"/>
<!--Match on application id with range as dictionary and integer-->
<Header appid="Sh,Accounting,16777251,Credit-Control" flags.r="True"/>

<!--Match on Credit Control with AVP content match-->
<Header appid="Credit-Control" flags.r="True">
  <AVP name="Destination-Realm" contents="squire.com"/>
</Header>
```

3.5.3 AVP-Match

The AVP-Match element contains a description of the AVP that needs to be matched.

The Message-Match element syntax is <AVP>

Attribute	Type	Options	Description
name	Attribute String		Either contains the Dictionary name of the AVP or the integer value
Flags.v	Attribute Boolean		Boolean value for the vendor specific bit in the AVP flags False : 0 True : 1
Flags.m	Attribute Boolean		Boolean value for the mandatory bit in the AVP flags False : 0 True : 1
Flags.p	Attribute Boolean		Boolean value for the protected bit in the AVP flags False : 0 True : 1
Length	Attribute Integer		Integer
vendorId	Attribute String		Either contains the Directory name of the vendor id or the integer value
Type	Attribute ENUM		Contains the type of the AVP Integer32 Interger64 Unsigned32 Unsigned64 Float32 Float64 IPV4Address IPV6Address OctectString UTF8String
Contents	Attribute String		Contains the value which the incoming AVP will be compared against.
regexp	Attribute String		For ASCII string based AVPs this can contain a regular expression that can be matched against.
Index	Attribute Value		Contains the index of the AVP for multiple instances of the same AVP.
Op	Attribute ENUM		Contains a mathematical operational field to allow for comparison of the defined contents with the incoming AVP's contents. See Operations
AVP	Element	Optional Repeatable	Used to identify the contents of a Grouped AVP. See AVP-Match
Contains	Element	Optional Repeatable	A list of contain elements that the contents of the AVP must match See AVP-Contains

3.5.3.1 AVP-Contains

The AVP-Contains element contains the content match of the parent AVP.

The AVP-Contains element syntax is <Contains>

Attribute	Type	Options	Description
Contents	Attribute String		Contains the value which the incoming AVP will be compared against.
Regexp	Attribute String		For ASCII string based AVPs this can contain a regular expression that can be matched against.

3.5.3.2 Examples

```
<!--Match on AVP with dictionary name if AVP is present-->
```

```
<AVP name="Destination-Host" op="present"/>
<!--Match on AVP with dictionary name if AVP is not present-->
<AVP name="Destination-Host" op="NotPresent"/>

<!--Match on OctetString AVP if contents match-->
<AVP name="Destination-Host" contents="squire.com"/>
<!--Match on OctetString AVP if contents do not match-->
<AVP name="Destination-Host" contents="squire.com" op="NotEqual"/>
<!--Match on OctetString AVP if contents match regular expression-->
<AVP name="User-Id" regexp="[0-9]{15}"/>

<!--Match on Unsigned32 AVP if contents match value-->
<AVP name="Result-Code" contents="2001"/>
<!--Match on Unsigned32 AVP if contents match range value-->
<AVP name="Result-Code" contents="3001-3004,3009"/>
<!--Match on Unsigned32 ENUM AVP-->
<AVP name="Result-Code" contents="DIAMETER_SUCCES"/>
<!--Match on Unsigned32 ENUM AVP list-->
<AVP name="Result-Code">
  <Contains contents="DIAMETER_COMMAND_UNSUPPORTED"/>
  <Contains contents="DIAMETER_UNABLE_TO_DELIVER"/>
  <Contains contents="DIAMETER_REALM_NOT_SERVED"/>
  <Contains contents="DIAMETER_TOO_BUSY"/>
</AVP>

<!--Match if second instance of AVP is present-->
<AVP name="Result-Code" index="2" op="present"/>

<!--Match if AVP flags are set as defined-->
<AVP name="Result-Code" flags.v="True" flags.m="False" flags.p="True"/>

<!--Match on contents of a Grouped AVP-->
<AVP name="LCS-Information">
  <AVP name="LCS-Client-ID">
    <AVP name="LCS-Client-External-ID" contents="aabbccddeeff"/>
    <AVP name="LCS-Client-Name">
      <AVP name="LCS-Name-String" contents="namestring"/>
    </AVP>
```

```
    </AVP>
    <AVP name="3GPP-IMSI" contents="250987654321"/>
  </AVP>
```

3.5.4 Peer-Match

The Peer-Match element allows for matching against names of the configured peers in the SVI-DSC. Depending on the Trigger the incoming peer and/or the outgoing peer can be defined. This is mainly used in the routing triggers allowing for routing actions to be defined depending on an incoming peer.

This can be useful in other triggers where a mediation list is shared across multiple peers

The following table indicates if the incoming or outgoing peers are available to match on:

Trigger	Incoming Peer	Outgoing Peer
IRP	✓ (Triggering Peer*)	✗
ORP	✓	✓ (Triggering Peer*)
IAP	✓ (Triggering Peer*)	✓
OAP	✓	✓ (Triggering Peer*)
RBR	✓	✗
RAR	✓	✓

For example: if Peer-Match is included on IRP, when IRP is triggered, the only peer available is the one the message arrived on: so Incoming peer is available, Outgoing peer is not.

* Triggering Peer in following example means:

```
<Mediate-Group TP="_TP_ORP">
...
<Peer-Match og-peer="HLR">
```

“HLR” MUST be the peer this trigger is attached to.

The Peer-Match element syntax is `<Peer-Match>`

Attribute	Type	Options	Description
ic-peer	Attribute String		Contains a single entry or comma separated entry of SVI-DSC peer names for the incoming peer match
og-peer	Attribute String		Contains a single entry or comma separated entry of SVI-DSC peer names for the outgoing peer match

3.5.5 Carrier-Match

The Carrier-Match element defines a MCC/MNC string to compare against the loaded Carrier-List to match access to or from a given carrier based on the MCC/MNC is allowed. The AVP which the MCC/MNC code is loaded from can either be determined from a previously stored variable of from a Save-Contents child variable defined in this element

The Carrier-Match element syntax is `<Carrier-Match>`

Attribute	Type	Options	Description
List	Attribute String		Specifies the name of the Carrier-List element defined in the Carriers Element
Key	Attribute String		Contains a complex regular expression that forms the MCC/MNC from the loaded variable. The order of the key must be MCCMNC. See Forming-Complex-Contents
Save-Contents	Element	Optional	This element can be used to load into a named variable the AVP that will be used to form the MCC/MNC key See Save-Contents

Example

The following example takes the MCC and MNC from a 3GPP formed address in the Destination-Realm AVP and does a lookup in Carrier-List “Partners” to see if a match is made

```
          <Carrier-Match list="Partners" key="${a.regex[4]}${?a.regex[2]}${a.regex[3]}">
            <Save-Contents name="Destination-Realm" regexp="mnc([0]|([1-9]))([0-9]{2}).mcc([0-
9]{3})" var="a"/>
          </Carrier-Match>
```

3.6 Actions

3.6.1 Action-Modify

The Action-Modify element allows for a Diameter message to be modified.

The Actions element syntax is `<Action-Modify>`

Attribute	Type	Options	Description
Store	Element	Optional	This contains a list of instructions to store the contents of an AVP to a variable to be used to data fill in the modify element. See Store
Modify	Element	Optional	If the diameter message is to be modified then this element contains the instructions to modify the message See Modify

3.6.2 Action-Ignore

The Action-Ignore element allows for a Diameter message to be dropped.

The Actions element syntax is `<Action-Modify>`

Attribute	Type	Options	Description

3.6.3 Action-Reject

The Action-Reject element allows for an incoming Diameter message to be rejected with a defined list of AVPS contained in the returned Answer message.

The reject message contains the following header and AVPs as standard

- Session-Id : If present in incoming message
- Origin-Realm: Incoming request peer's realm name
- Origin-Host : incoming request peer's host name
- Origin-State-Id : incoming request peer's current Origin-State-Id
- Error-Reporting-Host: incoming request peer's host name

Any additional AVPS can be added to the message by using the store and modify elements

The Actions element syntax is `<Action-Reject>`

Attribute	Type	Options	Description
Store	Element	Optional	This contains a list of instructions to store the contents of an AVP to a variable to be used to data fill in the modify element. See Store
Modify	Element	Optional	If the diameter message is to be modified then this element contains the instructions to modify the message See Modify

3.6.4 Action-Route

The Action-Route element allows for a Diameter message to be modified and the SVI-DSC routing table to be identified to be used for outgoing routing.

The Actions element syntax is `<Action-Route>`

Attribute	Type	Options	Description
Store	Element	Optional	This contains a list of instructions to store the contents of an AVP to a variable to be used to data fill in the modify element. See Store
Modify	Element	Optional	If the diameter message is to be modified then this element contains the instructions to modify the message See Modify
Route	Element	Mandatory	This contains the routing information for routing of the Diameter request See Route

3.6.5 Store

The Store element allows for the contents of AVPs to be stored against a variable name.

The Store element syntax is `<Store>`

Attribute	Type	Options	Description
AVP	Element	Repeatable Optional	This provides a path into an AVP group See Store-AVP
Save-Contents	Element	Repeatable Optional	This identifies the AVP's contents that is to be saved See Save-Contents
Save-AVP	Element	Repeatable Optional	This identifies the AVP that is to be saved See Save-AVP
Save-IMSI	Element	Repeatable Optional	This allows for String AVP containing a fully formed IMSI (MCC[3 digits]-MNC[2 or 3 Digits]-MSIN[9 or 10 digits]) to be stored as a variable which can be accessed as a variable. See Save-IMSI

3.6.5.1 Store-AVP

The Store-AVP element is used to provide a path for Grouped AVPs

The Store-AVP element syntax is `<AVP>`

Attribute	Type	Options	Description
Name	Attribute String		Contains the Dictionary name of the AVP
Index	Attribute Value		Contains the index of the AVP for multiple instances of the same AVP.
Save-Contents	Element	Repeatable Optional	This identifies the AVP's contents that is to be saved See Save-Contents
Save-AVP	Element	Repeatable Optional	This identifies the AVP that is to be saved See Save-AVP
AVP	Element	Repeatable	This provides a path into an AVP group See Store-AVP

3.6.5.2 Save-Contents

The Save-Contents element is used store the contents of an AVP against a variable name. This cannot be a grouped AVP.

The Save-Contents element syntax is `<Save-Contents>`

Attribute	Type	Options	Description
name	Attribute String		Contains the Dictionary name of the AVP
regexp	Attribute String		For ASCII string based AVPs this can contain a regular expression substitution syntax
Index	Attribute Value		Contains the index of the AVP for multiple instances of the same AVP.
Var	Attribute String		This contains the name of the variable for the contents to be saved against

3.6.5.3 Save-AVP

The Save-AVP element allows for the saving against a variable name of the identified AVP. This can be a Grouped AVP

The Save-AVP element syntax is `<Save-AVP>`

Attribute	Type	Options	Description
Name	Attribute String		Contains the Dictionary name of the AVP
Index	Attribute Value		Contains the index of the AVP for multiple instances of the same AVP.

Var	Attribute String		This contains the name of the variable against which an AVP is saved.

3.6.5.4 Save-IMSI

The Save-IMSI element is used store the contents of an AVP against a variable name of type IMSI. This cannot be a grouped AVP. The contents of the AVP must contain a string of MCC[3 digits],MNC[2 or 3 digits].MSIN[9 or 10 digits]

The Carrier list must be loaded into the SVI-DSC for this to succeed. To apply this variable in a complex format the extensions to the named variable are var.mcc, var.mnc, var.msin

The Save-IMSI element syntax is `<Save-IMSI>`

Attribute	Type	Options	Description
name	Attribute String		Contains the Dictionary name of the AVP
Index	Attribute Value		Contains the index of the AVP for multiple instances of the same AVP.
Var	Attribute String		This contains the name of the variable for the contents to be saved against

3.6.5.5 Examples

```
<!--Store contents from Grouped AVP and store a Grouped AVP-->

  <Store>

    <AVP name="LCS-Information">

      <AVP name="LCS-Client-ID">

        <Save-Contents name="LCS-Client-External-ID" var="lcs-client"/>

        <AVP name="LCS-Client-Name">

          <Save-Contents name="LCS-Name-String" var="lcs-name"/>

          <Save-IMSI name="User-Name" var="imsi"/>

        </AVP>

      </AVP>

      <Save-Contents name="3GPP-IMSI" regexp="([0-9]{3})[0-9]{3)[0-9]{9}" var="imsi"/>

    </AVP>

    <AVP name="PS-Information">

      <Save-AVP anme="MMS-Information" var="mmsinfo-avp"/>

    </AVP>

  </Store>



<!--Store contents from Grouped AVP and match with another Grouped AVP contents; then reject
based on comparison of the two -->

     <Rule name="get-first-number">

       <Action-Modify>

                 <Store>

                   <AVP name="Service-Information" >

                    <AVP name="MMS-Information" >

                         <AVP name="Recipient-Address">

                              <Save-Contents name="Address-Data" regexp="[0-9]" var="first"/>

                         </AVP>

                    </AVP>

                   </AVP>

                 </Store>

           </Action-Modify>

     </Rule>


      <Rule name="get-second-number">

           <Matches>

             <Message-Match>
```

```
                <AVP name="Subscription-Id">
                    <AVP name="Subscription-Id-Data" contents="$first" />
            </AVP>
          </Message-Match>
       </Matches>
       <Action-Reject>
      <Modify>
        <Add-AVP name="Result-Code" contents="2001"/>
                <Add-AVP name="Auth-Application-Id" contents="4"/>
                <Add-AVP name="CC-Request-Type" contents="4"/>
                <Add-AVP name="CC-Request-Number" contents="0"/>
      </Modify>
    </Action-Reject>
     </Rule>
```

3.6.6 Modify

The Modify Element provides instructions on how to modify the Diameter message

The Modify element syntax is `<Modify>`

Attribute	Type	Options	Description
AVP	Element	Repeatable Optional	This provides a path into an AVP group See Mod-AVP
Add-AVP	Element	Repeatable Optional	This adds an AVP to the end of a message or an identified Grouped AVP See Add-AVP
Remove-AVP	Element	Repeatable Optional	This removes an AVP from the message or an identified Grouped AVP. See Remove-AVP
Modify-AVP	Element	Repeatable Optional	This modifies an existing AVP if found in the message or identified Group AVP See Modify-AVP
Insert-AVP	Element	Repeatable Optional	This inserts an AVP into an identified position within a message or identified Group AVP See Insert-AVP
Modify-Header	Element	Optional	This allows for the modification of the diameter header. See Modify-AVP

3.6.6.1 Mod-AVP

The Mod-AVP element is used to provide a path for Grouped AVPs

The Mod-AVP element syntax is `<AVP>`

Attribute	Type	Options	Description
name	Attribute String		Contains the Dictionary name of the AVP
Index	Attribute Value		Contains the index of the AVP for multiple instances of the same AVP.
Add-AVP	Element	Repeatable Optional	This adds an AVP to the end of a message or an identified Grouped AVP See Add-AVP
Remove-AVP	Element	Repeatable Optional	This removes an AVP from the message or an identified Grouped AVP. See Remove-AVP
Modify-AVP	Element	Repeatable Optional	This modifies an existing AVP if found in the message or identified Group AVP See Modify-AVP
Insert-AVP	Element	Repeatable Optional	This inserts an AVP into an identified position within a message or identified Group AVP See Insert-AVP

3.6.6.2 Modify-Header

The Modify-Header element allows for the modification of the incoming message

The Modify-Header element syntax is `<Modify-Header>`

Attribute	Type	Options	Description
Code	Attribute String	Optional	If set this will change the code of the header
appid	Attribute String	Optional	If set this will change the vendor identity of the header
Flags.e	Attribute Boolean	Optional	If set this will change the value of the error flag of the header
Flags.t	Attribute Boolean	Optional	If set this will change the value of the re-transmission flag of the header
Flags.r	Attribute Boolean	Optional	If set this will change the of the request flag of the header
Version	Attribute Integer	Optional	If set this will change the value of the version field of the header

Examples

```
<!--Change header flag.e-->
<Modify-Header flags.e="True"/>
<!--Change header code and add-id.-->
<Modify-Header code="Credit-Control" appid="Credit-Control"/>
```

3.6.6.3 Add-AVP

The Add-AVP element will add an AVP to the end of a message or the end of a Grouped AVP.

The Add-AVP element syntax is <Add-AVP>

Attribute	Type	Options	Description
Present	Attribute ENUM	Optional	This describes the action which should be performed if the AVP is already present in the message Add: If the AVP is present then a new AVP will be added. Modify: If the AVP is present then the existing AVP is overwritten with the new contents Ignore: If the AVP is present then the AddAVP instruction will be ignored. Default: Add
Name	Attribute String Mandatory	Mandatory	Either contains the Dictionary name of the AVP or the integer value
Contents	Attribute String		Contains the new value of the AVP
Complex	Attribute String		The complex attribute allows for ASCII based AVPs to use a formula to use stored variables to form the contents. See Forming-Complex-Contents
Add-AVP	Element		For a Grouped AVP this contains the contents of this AVP. See Add-AVP

Examples

```
    <!--Add OctetStringAVP from contents-->
  <Add-AVP name="Destination-Host" contents="squire.com"/>
  <!--Add OctetStringAVP from previously stored variable lcs-name-->
  <Add-AVP name="Destination-Host" complex="${lcs-name}"/>
  <!--Add OctetStringAVP from previously stored regular expression variable imsi-->
  <Add-AVP name="Destination-Host"
complex="hss1.mnc+${imsi.regexp[1]}.mcc${imsi.regexp[2]}.3gppnetwork.org"/>


  <!--Add new AVP Group-->
  <Add-AVP name="LCS-Information">
    <Add-AVP name="LCS-Client-ID">
      <Add-AVP name="LCS-Client-External-ID" contents="lcs-client"/>
      <Add-AVP name="LCS-Client-Name">
        <Add-AVP name="LCS-Name-String" complex="${lcs-name}"/>
      </Add-AVP>
    </Add-AVP>
    <Add-AVP name="3GPP-IMSI" complex="127+${imsi.regexp[2]}${imsi.regexp[3]}"/>
  </Add-AVP>


  <!--Add AVP Contents to existing AVP group-->
  <AVP name="LCS-Information">
```

```
    <AVP name="LCS-Client-ID">
      <Add-AVP name="LCS-Client-External-ID" contents="lcs-client"/>
      <Add-AVP name="LCS-Client-Name">
        <Add-AVP name="LCS-Name-String" complex="${lcs-name}"/>
      </Add-AVP>
    </AVP>
    <Add-AVP name="3GPP-IMSI" complex="127+${imsi.regexp[2]}${imsi.regexp[3]}"/>
  </AVP>
```

3.6.6.4 Remove-AVP

The Remove-AVP element will remove if present the identified AVP from the diameter message.

The Remove-AVP element syntax is <Remove-AVP>

Attribute	Type	Repeat	Description
Name	Attribute String Mandatory		Either contains the Dictionary name of the AVP or the integer value of the AVP to be removed
Index	Attribute Value		Contains the index of the AVP for multiple instances of the same AVP.

Examples

```
<!--Remove AVP if present-->
<Remove-AVP name="Destination-Host"/>
<!--Remove 2nd instance of AVP if present-->
<Remove-AVP name="Result-Code" index="2"/>
<!--Remove Group AVP from within Group AVP-->
<AVP name="LCS-Information">
  <Remove-AVP name="LCS-Client-ID"/>
</AVP>
```

3.6.6.5 Modify-AVP

The Modify-AVP element will modify an AVP if present. Only one Modify-AVP element is allowed for a unique AVP

The <Modify-AVP> element syntax is `<Modify-AVP>`

Attribute	Type	Options	Description
Name	Attribute String Mandatory	Mandatory	Either contains the Dictionary name of the AVP or the integer value
Code	Attribute Integer	Optional	New Value of AVP code
Flags.v	Attribute Boolean		Boolean value for the vendor specific bit in the AVP flags to be changed to False : 0 True : 1
Flags.m	Attribute Boolean		Boolean value for the mandatory bit in the AVP flags to be changed to False : 0 True : 1
Flags.p	Attribute Boolean		Boolean value for the protected bit in the AVP flags to be changed to False : 0 True : 1
vendorId	Attribute String		Either contains the Directory name of the vendor id or the integer value to be changed to.
Contents	Attribute String		Contains the new value of the AVP
Complex	Attribute String		The complex attribute allows for ASCII based AVPs to use a formula to use stored variables to form the contents. See Forming-Complex-Contents

Examples

```
  <!--Modify AVP flags.m if present-->
  <Modify-AVP name="Destination-Host" flags.m="True"/>
  <!--Modify AVP contents-->
  <Modify-AVP name="Destination-Host" contents="squire.com"/>
  <!--Modify AVP contents from previously stored varible-->
  <Modify-AVP name="Destination-Host"
complex="hss1.mnc+${imsi.regexp[1]}.mcc${imsi.regexp[2]}.3gppnetwork.org"/>
```

3.6.6.6 Insert-AVP

The Insert-AVP element will add an AVP into a message or a Grouped AVP at the position defined by this element. If the position identified is to be after an AVP within the message and this AVP is not found, then the element can be configured to either add the AVP at the end of the message or AVP group or not to add the AVP. This cannot be used for a Group AVP

Attribute	Type	Options	Description
Position	Attribute String	Optional	Identifies where the AVP is to be inserted within the message or Grouped AVP. If After is selected then the Insert-Position must be provided Values: Top : Add to the top of the message or Grouped AVP

			End: Add to the end of the message or Grouped AVP After: Add after the AVP defined in the AVP-Position element
Nopresentadd	Attribute Boolean	Optional	If the position attribute is set to After and the AVP identified in the Insert-Position is not present in the message and this attribute is set to True, then the AVP will be added to the end of the message or AVP group.
Insert-Position	Element	Optional	Contains the AVP path for the location where the new AVP will be inserted See Insert-Position
Insert-Action		Optional	Defines the AVP to be added See Insert-Action

3.6.6.6.1 Insert-Position

This defines the AVP within the Diameter message at which the new AVP will be added

Attribute	Type	Options	Description
AVP	Element	Optional	AVP path for where the new AVP is inserted See Match-AVP

3.6.6.6.2 Insert-Action

This defines AVPs that are to be added within the Diameter message at the required position

Attribute	Type	Options	Description
Add-AVP	Element	Optional	AVP path for where the new AVP is inserted See Add-AVP

Example

```
<Insert-AVP>
   <Insert-Position>
     <AVP name="Destination-Realm"/>
   </Insert-Position>
   <Insert-Action>
     <Add-AVP name="Destination-Host" complex="bss1.${realm}"/>
   </Insert-Action>
</Insert-AVP>
```

3.6.7 Route

This element will allow the system to identify Diameter Route Table entries for routing purposes. These entries have to be marked with DSSME option. Note: Entries with DSSME option set are not evaluated during Normal-Routing process. Their role is to define routing for DSSME mediation engine Route action.

Attribute	Type	Options	Description
Name	Attribute String Mandatory	Mandatory	Contains the name of the Diameter Routing Table entry. In case of a redundant system, a route on each plane must be specified (as each plane has its own stack).

Example

```
<Rule name="route">
  <Action-Route>
    <Route>
      <Routing-Table name="pref_carrier1"/>
      <Routing-Table name="npref_carrier1"/>
    </Route>
  </Action-Route>
</Rule>
```

3.7 Operations

The following Operations can be used against AVP matches depending on the AVP type.

Value	Description
Equals	The result is true if the incoming value equals the defined value
NotEqual	The result is true if the incoming value does not equal the defined value
Greater	The result is true if the incoming value is greater than the defined value Not valid for strings or ranges
GreaterEquals	The result is true if the incoming value is greater or equal to the defined value. Not valid for strings or ranges
Smaller	The result is true if the incoming value is smaller than the defined value Not valid for strings or ranges
SmallerEquals	The result is true if the incoming value is smaller or equal to the defined value. Not valid for strings or ranges
Present	The result is true if the AVP is present
NotPresent	The result is true is the AVP is not present

3.8 Forming-Complex-Contents

It can be desirable to build an AVP from existing content from different AVPs within an incoming message. The user may also want to break down an AVP's content using regular expressions to add parts of this AVP into the new AVP's contents.

This is achieved by using the Save-AVP element to either save the entire contents string, or break down the contents via a regular expression. Once the contents have been saved the new contents can be constructed from the variables using the following syntax

(string | ${var})(string | ${var})…

The ${var} indicates the name of the variable that is to be used to be inserted into the contents. The string is an ASCII string.

3.8.1 Regular Expression Variable Type Substitution

If a regular expression with substitution/group syntax has been used in the Save-AVP element, these substitutions can be formatted into the contents by extending the variable syntax to var.regexp[n], where n is the substitution/group number from the regular expression.

If a regular expression group is not present when formatting the complex string the operation will fail and the new AVP will not be added to the modified message. This operation can be overridden by using the '?' character in front of the variable name to indicate this this variable is optional.

3.8.2 IMSI Type Substitution

If a variable has been loaded using the <Save-Imsi> element the variable's contents can be accessed using the three following type name, var.mcc, var.mnc and var.msin.

3.8.3 Examples

Take Destination-Realm contents and add Destination-Host with prefix with hss1.

```
<Actions>
  <Store>
    <Save-Contents name="Destination-Realm" var="var1"/>
  </Store>
  <Modify>
    <Add-AVP name="Destination-Host" complex="hss1.${var1}"/>
  </Modify>
</Actions>
```

Take IMSI from User-Id AVP and form into MCC.MNC.MSIN and then Add destination-host with 3GPP formatted address.

```
<Actions>
  <Store>
    <Save-Imsi name="User-Id" var="imsi"/>
  </Store>
  <Modify>
    <Add-AVP name="Destination-Host" complex="epc.mnc${var1.imsi.mnc}mcc${imsi.mcc}.3gppnetwork.org"/>
  </Modify>
</Actions>
```

4 Carrier List

The Carrier List allows for a list of carrier information including MCC and MNC to be loaded into the SVI-DSC to be used for the following purposes

- Create-White and Blacklists against AVP MCC/MNC information to determine if different actions should be undertaken due to destination of origination of transaction
- Allow for formatting of AVPS from an IMSI that may contain a 2 or 3 digit MNC

The top level of the Carrier list is the Carriers element.

4.1 Carriers

This element is the top level of the carrier list.

Attribute	Type	Options	Description
Partner-List	Element	Optional	This element contains a list of carriers See Carrier-List

4.1.1 Partner-List

This element is the top level of the carrier list.

Attribute	Type	Options	Description
Name	Mandatory String		This contains a name identifier to be used by other elements when accessing the carrier list

Carrier	Element	Repeatable	This contains carrier information

4.1.2 Carrier

This element contains the information on the Carrier

Attribute	Type	Options	Description
Mcc	String	Mandatory	Contains the mcc of a carrier
Mnc	String	Mandatory	Contains the mnc of a carrier. For a 3 digit MNC this will have a length of 3. For a 3 digit MNC this will have a length of 2
Country	String	Optional	This optional field can contain the country the carrier resides in.
County-code	String	Optional	This contains the country code of the carrier
Network	String	Optional	This contains the carriers name