Reference Guide

SIP Message Manipulation

Mediant™ Series SBCs, Gateways & MSBRs

Version 7.0

HD VoIP Sounds Better

AudioCodes

Table of Contents

List of Tables

This page is intentionally left blank.

Notice

This document is a Reference Guide for AudioCodes SIP Message Manipulation feature.

Information contained in this document is believed to be accurate and reliable at the time of printing. However, due to ongoing product improvements and revisions, AudioCodes cannot guarantee accuracy of printed material after the Date Published nor can it accept responsibility for errors or omissions. Before consulting this document, check the corresponding Release Notes regarding feature preconditions and/or specific support in this release. In cases where there are discrepancies between this document and the Release Notes, the information in the Release Notes supersedes that in this document. Updates to this document and other documents as well as software files can be downloaded by registered customers at http://www.audiocodes.com/downloads.

Date Published: December-30-2015

Trademarks

WEEE EU Directive

Pursuant to the WEEE EU Directive, electronic and electrical waste must not be disposed of with unsorted waste. Please contact your local recycling authority for disposal of this product.

Customer Support

Customer technical support and services are provided by AudioCodes or by an authorized AudioCodes Service Partner. For more information on how to buy technical support for AudioCodes products and for contact information, please visit our Web site at www.audiocodes.com/support.

Abbreviations and Terminology

Each abbreviation, unless widely used, is spelled out in full when first used.

Document Revision Record

LTRT	Description
28623	Initial document release for Version 7.0.
28628	URL parameter update; document restructuring.
28629	Description of Row Rule field added; note re IPGroup_SIPGroupName added; Session variable (var.session) added; Unknown header example updated.

Documentation Feedback

AudioCodes continually strives to produce high quality documentation. If you have any comments (suggestions or errors) regarding this document, please fill out the Documentation Feedback form on our Web site at http://www.audiocodes.com/downloads. Your valuable feedback is highly appreciated.

1 Introduction

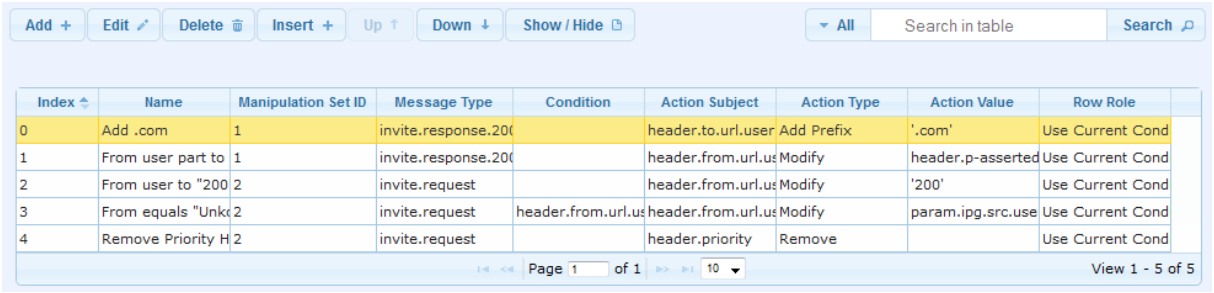
This document provides a reference guide with examples for configuring SIP message manipulation rules in the Message Manipulation table. It describes each field in the table and the supported syntax.

This page is intentionally left blank.

2 Message Manipulation Table Fields

SIP Message Manipulation is configured in the Message Manipulations table (**Configuration** tab > **VoIP** > **SIP Definitions** > **Msg Policy & Manipulation** > **Message Manipulations**). The figure below shows an example of SIP Message Manipulation rules in the table.

Figure 2-1: Message Manipulations Table



This section describes the Message Manipulation table fields and their syntax used for entering the values:

- Manipulation Set ID – see Section 2.1 on page 11
- Message Type – see Section 2.2 on page 12
- Condition – see Section 2.3 on page 13
- Action - See Section 2.4 on page 14
  - Action Subject
  - Action Type
  - Action Value
- Row Rule – see Section 2.5 on page 15

2.1 Manipulation Set ID

The 'Manipulation Set ID' field enables you to group message manipulation rules that you have defined. Once you have defined manipulation rules and associated them with a specific Manipulation Set ID, you **must** assign this ID to the relevant IP Group in the IP Group table, where they can be assigned to either the inbound (Inbound Message Manipulation Set) or outbound (Outbound Message Manipulation Set) leg.

Syntax:

```
<0-19>
```

where:

- <0-19> specifies the Manipulation Set ID.

2.2 Message Type

The following syntax determines the type of message to which the manipulation rule refers.

Syntax:

```
<SIP-method/any>.<request/response/any>.<response-type>
```

where:

- **<SIP-method/any>** specifies the SIP method used with the option to specify requests of all method types.
- **<request/response/any>** specifies the SIP request or SIP response type with the option to specify any request or response type.
- **<response-type>** specifies the SIP response type. You can also use the 'x' wildcard to denote multiple response types:
  - To denote all response types belonging to a specific response group (i.e., 1xx for provisional, 2xx for successful, 3xx for redirection, 4xx for client failure, 5xx for server failure, and 6xx for global failure responses), use two 'x' wildcards instead of the last two digits of the response: `<first digit of response group>xx` (e.g., `1xx`)

The following table provides examples of different message types.

Table 2-1: Message Type Examples

Message Types	Description
`invite.request`	INVITE requests
`invite.response.200`	INVITE 200 responses only
`register.response.2xx`	All 2xx responses for REGISTER
`subscribe.request`	All SUBSCRIBE requests
`subscribe.response`	All SUBSCRIBE responses
`reinvite.request`	re-INVITE requests
`any.request`	Requests of all method types, where any is a keyword.
`any.response.200`	All 200 responses for all method types, where any is a keyword.
`invite`	Requests and responses of INVITE method.
`<empty>`	All request and responses for all method types.
`info.any`	All INFO requests and responses.
`private1.request`	All requests with method 'private1'.

2.3 Condition

The 'Condition' field is used to test specific parts of the header in the message with specified values. Conditions may be combined with other conditions using logical operators (and/or).

Syntax:

```
<subject> <operand> <value>
```

where:

- **<subject>** specifies the subject of the condition using the following format: header/body/parameter
- **<operand>** specifies the operand of the condition using the following format: condition-operand
- **<value>** specifies the value of the condition using the following format: string/header/body/parameter/random/variable/regex

The following table provides various examples of different conditions.

Table 2-2: Condition Examples

Condition	Description
`header.expires.time < '88888'`	Returns true if expires time is less than '88888'.
`header.user-agent contains 'Android-VMAS' OR header.user-agent contains 'MP252'`	Returns true if the user agent is 'Android-VMAS' or 'MP252'.
`param.message.sdp.address == '10.132.10.101'`	Returns true if the "c=" line contains the given IP address.
`header.request-uri.methodtype=='415'`	Returns true if the message method type is '415'.
`header.diversion.0 regex (<.*)(;urlparam=[a-z]*)(.*>)`	Returns true if the REGEX engine matches urlparam=<specific value>.

2.4 Action

The following describes the syntax of the 'Action' field:

Syntax:

```
<Action Subject>
```

where:

- **<Action Subject>** specifies the message component upon which you wish to manipulate, using the following format:

  header/body/variable

Syntax:

```
<Action Type>
```

where:

- **<Action Type>** specifies the type of action you wish to perform on the message component, using the following format:

  action-operand

Syntax:

```
<Action Value>
```

where:

- **<Action Value>** specifies the value to assign to the Action Type and Action Subject, using the following format:

  string/header/body/parameter/random/variable/regex

The following table provides various example actions.

Table 2-3: Action Examples

Action Subject	Action Type	Action Value	Description
`header.customername`	**Add**	`'Audiocodes'`	Adds the "customername" header to the message with a value of "Audiocodes".
`header.customername`	**Delete**		Deletes the header "customername" from the message.
`var.global.0`	**Modify**	`header.user-agent.content`	Stores the content of the User-agent header in a global variable. Note, the **Modify** action is executed on the variables (not the **Add** action).
`header.contact.param.company`	**Add**	`'audiocodes'`	Adds a parameter "company" to a Contact header and assigns the value "Audiocodes" to it.

2.5 Row Rule

The 'Row Rule' field determines which Condition (configured in the 'Condition' field) the rule uses. The rule can use the Condition configured for the rule itself or the Condition configured for a previous rule. Using the Condition of a previous rule allows you to configure multiple manipulation rules using the same condition.

- [0] Use Current Condition = (Default) The Condition configured for the rule itself (i.e., in the same table row) is used.
- [1] Use Previous Condition = The Condition configured in the first (closest) table row above the rule that is configured to **Use Current Condition** is used. For example, if Index 3 is configured to **Use Current Condition** and Index 4 and 5 are configured to **Use Previous Condition**, Index 4 and 5 use the condition configured for Index 3. The following figure shows a configuration example where Index 1 and 2 ('Row Rule' configured to **Use Previous Condition**) use the condition configured for Index 0 ('Row Rule' configured to **Use Current Condition**):

Figure 2-2: Configuration Example for Message Manipulation Rules using Same Condition

Index	Name	Manipulation Set ID	Message Type	Condition	Action Subject	Action Type	Action Value	Row Role
0	To header urgent	0	invite.request	header.request-uri.(	header.to	Modify	header.to + ':urgent=1'	Use Current Condition
1	Add emergency	0			header.priority	Add	'emergency'	Use Previous Condition
2	User-Agent	0			header.user-agent	Modify	'trunk-a'	Use Previous Condition

Note: When configured to **Use Previous Condition**, the 'Message Type' and 'Condition' fields are not applicable and if configured are ignored.

This page is intentionally left blank.

3 Detailed Field Syntax

This section describes the detailed syntax usage of the fields in the Message Manipulations table. The following syntax is described:

- **Condition Operands** – see Section 3.1 below.
- **Action Operands** – see Section 3.2 on page 18.
- **Strings** – see Section 3.3 on page 18.
- **Headers** – see Section 3.4 on page 19.
- **Body** – see Section 3.5 on page 28.
- **Parameters** – see Section 3.6 on page 29.

3.1 Condition Operands

The following table describes the condition operands.

Table 3-1: Condition Operands

Condition Operand	Description
== / !=	Tests for equivalent / not equivalent values.
>= / <=	Tests for greater than or equal to / less than or equal to values.
> / <	Tests for greater than / less than values.
contains / !contains	Tests a string containing / not containing specified text.
exists /!exists	Tests whether a parameter exists / does not exist.
Suffix / prefix	Tests whether a string has a particular suffix / prefix.
len> / len< / len==	Tests whether the length of a string is greater than / less than / equal to a specific value.
regex	Tests whether a string matches the given regular expression.

3.2 Action Operands

The following table describes the action operands.

Table 3-2: Action Operands

Action Operand	Description
Add	Adds entities to a message.
Remove	Removes entities from a message.
Modify	Modifies parts of a header or SDP.
Add Prefix	Adds a string prefix to part of a header.
Add Suffix	Adds a string suffix to part of a header.
Remove Prefix	Removes a string prefix from part of a header.
Remove Suffix	Removes a string suffix to part of a header.

3.3 Strings

The string type is the most basic of all syntax types. A string is a series of characters enclosed by single apostrophe. It can be used as the value for the following Message Manipulation table fields:

- Condition
- Action Value

The following table provides configuration examples for using strings in the Message Manipulations table.

Table 3-3: Examples of Using Strings

Message Type	Condition	Action Subject	Action Type	Action Value
`invite.request`	`header.user-agent.content contains 'X-Lite'`	`header.user-agent.content`	**`Modify`**	`'anonymous UA'`
`invite.request`	`header.from.url.user=='101;ext=7166'`	`header.user-agent.content`	**`Modify`**	`'anonymous UA'`

3.4 Headers

This section describes the syntax used for SIP headers in the Message Manipulations table.

Syntax:

```
header.<header-name>.<header-index>.<sub-type>
```

where:

- **<header-name>** specifies the header name as it arrives in the message. For example: From, To, Contact (not case sensitive).
- **<header-index>** refers to a specific header, in the event where more than one header of the same type is present in the message. The index starts at 0, therefore in order to refer to the first header in the list, the header-index value should be 0. For example, *header.contact.2* would refer to the third header in the list.
  If <header-index> is not specified; however, a <sub-type> exists, then the sub-type would reference the first header in the list, i.e. *header.contact.url.user* is identical to *header.contact.0.url.user*.

  If both <header-index> and <sub-type> are not specified, then the subject would refer to all headers of this type. For example, to remove or modify all headers of a specific type, refer to the header as *header.contact*.
- **<sub-type>** specifies a specific part of the message. For example, url.user, url.host etc.

Note: The SIP Group Name (IPGroup_SIPGroupName) parameter of the IP Group table overrides inbound message manipulation rules that manipulate the host name in Request-URI, To, and/or From SIP headers. If you configure a SIP Group Name for an IP Group and you want to manipulate the host name in these SIP headers, you must apply your manipulation rule (Manipulation Set ID) to the IP Group as an Outbound Message Manipulation Set (IPGroup_OutboundManSet), when the IP Group is the destination of the call. If you apply the Manipulation Set as an Inbound Message Manipulation Set (IPGroup_InboundManSet), when the IP Group is the source of the call, the manipulation rule is overridden by the SIP Group Name.

3.4.1 Detailed Header Syntax

The table below describes the syntax to manipulate the various SIP headers:

Table 3-4: Syntax for Manipulating SIP Headers

SIP Header	Attribute to Manipulate	Manipulation Syntax	Example
Accept	Header itself	`header.accept`	
Accept-Language	Header itself	`header.accept-language`	
Allow	Header itself	`header.allow`	
Call-Id	Header itself	`header.call-id`	
	Specific ID	`header.call-id.id`	
Contact	Header itself	`header.contact`	
	Expires	`header.contact.expires`	
	Globally Routable	`header.contact.gruucontact`	

SIP Header	Attribute to Manipulate	Manipulation Syntax	Example
	UA URI (GRUU) contact		
	Enable GRUU	`header.contact.isgruu`	
	Name	`header.contact.name`	
	Parameter	`header.contact.param`	
	URL	`header.contact.url.<url` Where <*url*> can be:	
		▪ type: Defines the type of URL: ✓ 1: Indicates a SIP URI (sip:) ✓ 2: Indicates a SIP Tel URI (tel:) ✓ 3: Indicates a fax URI (fax:) ✓ 4: Indicates a SIPS URI (sips:)	`header.contact.url.type == '1'`
		▪ host: Indicates host part. The host by itself includes both domain name/IP address and port, e.g., 10.33.2.6:5070. However, you can indicate only the name/IP address or only the port: ✓ name: Indicates the host name ✓ port: Indicates the port	`header.contact.url.host.port`
		▪ mhost: Indicates the SIP 'maddr' parameter (see RFC 3261)	
		▪ userphone: Indicates the SIP 'user=phone' parameter (the tel URI). (See Note below.)	`header.contact.url.userphone`
		▪ looseroute: Indicates loose routing parameter ('lr') according to the Record-Route set (see Note below)	
		▪ user: Indicates the user part of the URI (string)	`header.contact.url.user=='401'`
		▪ transporttype: ✓ 0: UDP ✓ 1: TCP ✓ 2: TLS ✓ 3: SCTP	`header.contact.url.transporttype == '0'`
		▪ param: Indicates a SIP parameter for the URI (can add, for example)	`header.contact.url.param.subject`
		Notes: ▪ For type, host, mhost, userphone, looseroute, user, and transporttype, the 'Action Type' field must be set to Modify. ▪ For userphone and looseroute, configure the rule with the 'Action Value' field set to '0' (to remove) or '1' (to add).	
Cseq	Header itself	`header.cseq`	
	Number	`header.cseq.num`	`header.cseq.num=='1'`
	Type	`header.cseq.type`	

SIP Header	Attribute to Manipulate	Manipulation Syntax	Example
Diversion	Header itself	`header.diversion`	
	Name	`header.diversion.name`	
	Parameter	`header.diversion.param`	
	Privacy - 1 (full) / 2 (off)	`header.diversion.privacy`	`header.diversion.privacy=='1'`
	Reason (enum)	`header.diversion.reason`	
	Screen – yes / no	`header.diversion.screen`	
	URL (see URL for Contact header)	`header.diversion.url`	
Event	Header itself	`header.event`	

	Event Key
	ID
	Event package
		`header.event.eventkey`
		`header.event.eventkey.id`
		`header.event.eventkey.eventpackage`	

	Parameter	`header.event.param`	`header.event.param.itsp-abc`
Expires	Header itself	`header.expires`	
	Expiry time	`header.expires.time`	
From	Header itself	`header.from`	
	Name	`header.from.name`	
	Remove quotation marks surrounding display name	`header.from.quotecontrol` The 'Action Value' field must be set to '0'.	
	Parameter	`header.from.param`	`header.from.param.p1`
	Tag	`header.from.tag`	
	URL (see URL for Contact header)	`header.from.url`	`header.from.url.user != '654'`
History-Info	Header itself	`header.history-info`	
Max-Forwards	Header itself	`header.max-forwards`	
	Value	`header.max-forwards.val`	
Min-Se and Min-Expires	Header itself	`header.min-se` `header.min-expires`	
	Parameter	`header.min-expires.param`	
	Time	`header.min-expires.time`	
P-Asserted-Identity	Header itself	`header.p-asserted-identity`	
	Name (string)	`header.p-asserted-identity.name`	
	URL (see URL for Contact header)	`header.p-asserted-identity.url`	`header.p-asserted-identity.url.host`

SIP Header	Attribute to Manipulate	Manipulation Syntax	Example
P-Associated-URI	Header itself	`header.p-associated-uri`	
	Name (string)	`header.p-associated-uri.name`	
	Parameter	`header.p-associated-uri.param`	
	URL (see URL for Contact header)	`header.p-associated-uri.url`	
P-Called-Party-ID	Header itself	`header.p-called-party-id`	
	Name (string	`header.p-called-party-id.name`	
	Parameter	`header.p-called-party-id.param`	`header.p-called-party-id.param.p1`
	URL (see URL for Contact header)	`header.p-called-party-id.url`	
P-Charging-Vector	Header itself	`header.p-charging-vector`	
P-Preferred-Identity	Header itself	`header.p-preferred-identity`	
	Name (string)	`header.p-preferred-identity.name`	
	URL (see URL for Contact header)	`header.p-preferred-identity.url`	
Privacy	Header itself	`header.privacy`	
	Privacy types	`header.privacy.privacy.<type>` where <type> can be: ▪ none ▪ header ▪ session ▪ user ▪ critical ▪ identity ▪ history	`header.privacy.privacy.user`
Proxy-Require	Header itself	`header.proxy-require`	
	SIP Capabilities	`header.proxy-require.<capability>` where <capability> can be: ▪ earlymedia ▪ reliableresponse ▪ timer ▪ earlysession ▪ privacy ▪ replaces ▪ history ▪ unknown ▪ gruu ▪ resourcepriority ▪ targetdialog ▪ sdpanat	`header.proxy-require.earlymedi`

SIP Header	Attribute to Manipulate	Manipulation Syntax	Example
Reason	Header itself	`header.reason`	
	Reason types	`header.reason.reason.<type>` where <type> can be: ▪ reason ▪ cause ▪ text	`header.reason.reason.reason`
	MLPP: Type: Preemption (0), MLPP (1) cause	`header.reason.mlpp`	
Referred-By	Header itself	`header.referred-by`	
	Parameter	`header.referred-by.param`	`header.referred-by.param.p1`
	URL (see URL for Contact header)	`header.referred-by.url`	`header.referred-by.url.host`
Refer-To	Header itself	`header.refer-to`	
Remote-Party-ID	Header itself	`header.remote-party-id`	
	Counter	`header.remote-party-id.counter`	
	Name	`header.remote-party-id.name`	
	Number Plan	`header.remote-party-id.numberplan` where <numberplan> can have the following value: ▪ 1: ISDN ▪ 3: Data ▪ 4: Telex ▪ 8: National ▪ 9: Private ▪ 15: Reserved	
	Number Type	`header.remote-party-id.numbertype`	
	Parameter	`header.remote-party-id.param`	
	Privacy (see Privacy header for description)	`header.remote-party-id.privacy`	
	Reason types	`header.remote-party-id.reason.<type>` where <type> can be: ▪ busy ▪ immediate ▪ no answer	`header.remote-party-id.reason.busy`
	Screen – Yes / No	`header.remote-party-id.screen`	
	Screen Indicator types	`header.remote-party-id.screening` where screening can equal the following enumeration value: ▪ -1: Screening not included ▪ 0: user provided ▪ 1: user passed ▪ 2: user failed	`header.remote-party-id.screening == 0`

SIP Header	Attribute to Manipulate	Manipulation Syntax	Example
		▪ 3: network provided	
	URL (see URL for Contact header)	`header.remote-party-id.url`	
Request-URI	Header itself	`header.request-uri`	
	Method	`header.request-uri.method`	
	Method Type	`header.request-uri.methodtype` The following enumerations are used to represent the SIP methods: ▪ 5: INVITE ▪ 6: re-INVITE ▪ 7: BYE ▪ 8: OPTIONS ▪ 9: ACK ▪ 10: CANCEL ▪ 11: REGISTER ▪ 12: INFO ▪ 13: MESSAGE ▪ 14: NOTIFY ▪ 15: REFER ▪ 16: SUBSCRIBE ▪ 17: PRACK ▪ 18: UPDATE ▪ 19: PUBLISH ▪ 21: SERVICE	`header.request-uri.methodtype == '5'` (i.e., SIP method is INVITE message)
	URI	`header.request-uri.uri`	
	URL (see URL for Contact header)	`header.request-uri.url`	`header.request-uri.url.user == '101'`
Require	Header itself	`header.require`	
	SIP Capabilities (see SIP Capabilities for Proxy-Require header)	`header.require`	`header.require.earlymedia`
Resource-Priority	Header itself	`header.resource-priority`	
	Namespace	`header.resource-priority.namespace`	
	RPriority	`header.resource-priority.rpriority`	
Retry-After	Header itself	`header.retry-after`	
	Time	`header.retry-after.time`	
Server or User-Agent	Header itself	`header.user-agent` `header.server`	
Service-Route	Header itself	`header.service-route`	
	Service route list entry	`header.service-route.<entry>.serviceroute`	`header.serviceroute.1.serviceroute`
Session-Expires	Header itself	`header.session-expires`	
	Parameter	`header.session-expires.param`	`header.session-`

SIP Header	Attribute to Manipulate	Manipulation Syntax	Example
			`expires.param.longtimer`
	Refresher	`header.session-expires.refresher`	**Note:** The Action Value '1' sets it to "UAC"; the value '2' sets it to "UAS" (i.e., UA type doing the refreshing)
	Time	`header.session-expires.time`	
Subject	Header itself	`header.subject`	
Supported	Header itself	`header.supported`	
	SIP Capabilities (see SIP Capabilities for Proxy-Require header)	`header.supported.<capability>`	`header.supported.path`
To	Header itself	`header.to`	
	Display name	`header.to.name`	
	Parameter	`header.to.param`	`header.to.param.artist`
	tag	`header.to.tag`	
	URL (see URL for Contact header)	`header.to.url`	`header.to.url.userphone`
Unsupported	Header itself	`header.unsupported`	
	SIP Capabilities (see SIP Capabilities for Proxy-Require header)	`header.unsupported.<capability>`	`header.unsupported.path`
User-To-User and X-UserToUser	Header itself	`header.x-usertouser`	
	User-to-User Descriptor	`header.x-usertouser.user2user`	
	Protocol Descriptor (PD)	`header.x-usertouser.pd`	

SIP Header	Attribute to Manipulate	Manipulation Syntax	Example
Via	Header itself	`header.via`	
	Alias	`header.via.alias`	
	Branch	`header.via.branch`	
	Host name	`header.via.host`	
	Via parameter 'maddr'	`header.via.maddrip`	
	Parameter	`header.via.param`	
	Port	`header.via.port`	
	Transport type	`header.via.transporttype` where transporttype can have the following value: ▪ 0: UDP ▪ 1: TCP ▪ 2: TLS ▪ 3: SCTP	`header.via.0.transporttype == '0'`
Warning	Header itself	`header.warning`	
Unknown headers	Header itself	`header.<unknown header name>`	`header.color`

3.4.2 Header Examples

The following table provides examples of syntax for indicating header fields.

Table 3-5: Header Field Syntax Examples

Header	Description
`header.to`	Defines the top level of the To header.
`header.to.url.user`	Defines the user part in the header SIP URL.
`header.from.url.host`	Defines the host part in the From header.
`header.from.name`	Defines the display name in the From header.
`header.newheader`	Defines a header *newheader*.
`header.contact.param.newparam`	Defines the parameter *newparam* of a Contact header.
`header.refer-to.url.host`	Defines the host part of the Refer-To header.
`header.diversion.reason`	Defines the Reason parameter in the Diversion header.
`header.supported.capabilities.path`	Defines the supported headers capabilities *path.*
`header.supported.capabilities.replaces`	Defines the supported headers capabilities *replaces*.
`header.max-forwards.val`	Defines the value of the Max-Forwards header.
`header.request-uri.methodtype`	Defines the method in the Request-URI.
`header.remote-party-id.0.partytype`	Defines the party type in the firts Remote-Party-ID header.
`header.contact.3`	Defines the third Contact header.
`header.via.2.url.user`	Defines the user part of the second Via header.

The following table provides examples of manipulation rules for headers.

Table 3-6: Header Field Manipulation Rules Examples

Message Type	Condition	Action Subject	Action Type	Action Value
`register.request`	`header.from.url.user == '101' OR header.from.url.user == '1000'`	`header.from.url.user`	**Modify**	`'2000'`
`register`		`header.to.url.host.name`	**Modify**	`'audiocodes.com'`
`invite`		`header.from.name`	**Modify**	`header.contact.url.user`
`invite.request`		`header.newheader`	**Add**	`'information to client'`
`subscribe`	`header.via.transporttype=='1'`	`header.to.param.transporttype`	**Add**	`'TCP'`

3.5 Body

This section describes the syntax used for the SIP body in the Message Manipulations table.

Syntax:

```
body.<body-name>
```

where:

<body-name> specified the body name as it arrives in the message. For example, 'application/sdp' (case-insensitive).

3.5.1 Body Examples

The following table provides examples of the syntax for indicating the SIP message body.

Table 3-7: Message Body Syntax Examples

Subject	Description
`body.application/x-nt-mcdn-frag-hex`	Adds or removes this 'unknown' body type.
`body.sdp`	Defines the SDP in the body.

The following table provides configuration examples of manipulation rules for the message body.

Table 3-8: Message Body Manipulation Rules Examples

Message Type	Condition	Action Subject	Action Type	Action Value
`invite`	`body.sdp !exists`	`body.application/ x-nt-mcdn-frag-hex`	**Add**	`'a=0981233\\b=12rew wer\\note=newlinecha racter'`
`invite.re quest`		`body.mwi`	**Add**	`'Messages-Waiting: yes\\Message-Account: sip:alice@vmail.exam ple.com\\Voice-Message: 2/8 (0/2)'`
`any`		`body.mwi.summary.ne wmsgs`	**Modify**	`'23'`
`invite`		`body.mwi.summary.ol dmsgs`	**Modify**	`'18'`
`invite`		`body.mwi.summary.ne wurgentmsgs`	**Modify**	`'12'`
`any`		`body.mwi.summary.ol durgentmsgs`	**Modify**	`'67'`
`invite`		`body.mwi.pending`	**Modify**	`'8'`
`invite`		`body.mwi.messagewai ting`	**Modify**	`'2'`

3.6 Parameters

This section describes the syntax used for the following SIP parameter types in the Message Manipulations table:

- Message Parameters
- IP Group Parameters
- Call Parameters

3.6.1 Message Parameter Syntax

The following table describes the syntax used for Message parameters in the Message Manipulations table.

Table 3-9: Message Parameter Syntax

Subject	Description
`param.message.sdp.address`	Specifies the address in the SDP.
`param.message.sdp.rtpmode`	Specifies the RTP mode in the SDP.
`param.message.sdp.originaddress`	Specifies the origin address in the SDP.
`param.message.sdp.port`	Specifies the port in the SDP.
`param.message.address.<src/dst>.port`	Specifies the port as a string for the source or destination of the message.
`param.message.address.<src/dst>.address`	Specifies the IP address as a string for the source or destination of the message.
`param.message.address.<src/dst>.<transporttype>`	Specifies the transport type as a string for the source or destination of the message. where **<transporttype>** is one of the following values: ▪ UDP ▪ TCP ▪ TLS

3.6.2 IP Group Parameter Syntax

The following table describes the syntax used for IP Group parameters in the Message Manipulations table.

Table 3-10: IP Group Parameter Syntax

Subject	Description
`param.ipg.<src/dst>.user`	Specifies the source or destination contact address for an active call.
`param.ipg.<src/dst>.host`	Specifies the source or destination IP Group name for an active call.
`param.ipg.<src/dst>.type`	Specifies the source or destination IP Group type for an active call. where <src/dst> is one of the following values: ▪ Server ▪ User ▪ Gateway
`param.ipg.<src/dst>.id`	Specifies the source or destination IP Group ID as a string for an active call.
`param.ipg.<src/dst>.user-defined.<0-1>`	Specifies the source or destination IP Group's user-defined string for manipulation rules in the IP Group table, where: ▪ *0* uses the string configured for the IPGroup_MsgManUserDef1 parameter in the IP Group table ▪ *1* uses the string configured for the IPGroup_MsgManUserDef2 parameter in the IP Group table

3.6.3 Call Parameter Syntax

The following table describes the syntax used for Call parameters in the Message Manipulations table.

Table 3-11: Call Parameter Syntax

Subject	Description
`param.call.<src/dst>.user`	Specifies the source or destination username during run-time.
`param.call.<src/dst>.nat`	Enables manipulation of a SIP message depending on whether (=='true') or not (=='false') the source or destination of the message is located behind NAT. The keywords can be used in the 'Condition' or 'Action Value' parameters in the Message Manipulations table. Message Manipulation rules using the keywords are applicable only to message manipulation on the outbound leg (i.e., the rules can only be assigned to the 'Outbound Message Manipulation Set' parameter in the IP Group table.

3.6.4 Parameter Examples

The following table provides configuration examples for using parameters in the Message Manipulations table.

Table 3-12: Call Parameter Examples

Message Type	Condition	Action Subject	Action Type	Action Value	Description
	`param.message.sdp.address == '10.132.10.101'`	`header.IPSource`	**Add**	`param.ipg.src.id`	If the address in the SDP is 10.132.10.101, the <device> adds a new SIP header, "IPSource" whose value is set to the ID of the source IP Group
`invite.response.200`	`param.message.sdp.rtpmode=='inactive'`	`header.origin`	**Add**	`param.message.sdp.originaddress`	In 200 OK messages, if the RTP mode is inactive, add a new header, "origin" whose value is set to the address in the origin ('o=') SDP
	`param.message.sdp.rtpmode== 'inactive'`	`header.from.param.origin`	**Add**	`param.message.sdp.originaddress`	If the RTP mode is inactive, add a new parameter, "origin" to the From header. The value of the parameter is set to the 'o=' address in the SDP.
`subscribe.request`		`header.to.param.user`	**Add**	`param.call.src.user`	In SUBSCRIBE messages, add the parameter, "user" to the To header. The value is set to the source username.
`invite.response`		`header.request-uri.url.param.myname`	**Add**	`param.ipg.src.host`	For INVITE responses, add the myname parameter to the Request-URI. The parameter value is taken from the 'Group Name' field of the IP Group.
`invite`		`header.MyCustomHeader`	**Add**	`param.ipg.dst.user-defined.0`	For INVITE messages, add a header called "MyCustomHeader" and whose value is taken from the IPGroup_MsgManUserDef1 field in the IP Group.
`any.request`		`header.session-expires.refres`	**Modify**	`'1'`	Manipulates the 'refresher' parameter to "UAC" in the

Message Type	Condition	Action Subject	Action Type	Action Value	Description
		her			Session-Expires header (i.e., UAC is doing the refreshing). For example: Session-Expires: 180;refresher=uac
invite	param.message.sdp.rtpmode=='sendonly' and param.call.dst.nat=='true'	param.message.sdp.rtpmode	**Modify**	'sendrecv'	If the device determines that the destination of the INVITE message is located behind NAT (param.call.dst.nat=='true'), and the RTP mode in the SDP of the incoming INVITE is 'sendonly' (param.message.sdp.rtpmode=='sendonly'), it changes the RTP mode to 'sendrecv' in the SDP of the outgoing INVITE.

4 Advanced Manipulation Features

This chapter describes advanced features that you can use for manipulating SIP messages.

4.1 Wildcards for Header Removal

The device supports the use of the "*" wildcard character to remove headers. The "*" character may only appear at the end of a string. For example, "X-*" is a valid wildcard request, but "X-*ID" is not.

Below are examples of using the wildcard:

- header.p-* - removes all headers that have the prefix "p-"
- header.x-vendor* - removes all headers that start with "x-vendor"

Note: The wildcard does not remove the following headers:

- Request-Uri
- Via
- From
- To
- Callid
- Cseq
- Contact

4.2 Random Characters

The following syntax shows how to specify random letter characters in the range *a* to *z* in Message Manipulation rules.

Syntax:

```
rand.string.<n>.a.z
```

where:

- **<n>** is the number of random letter characters you wish to specify in the range *a* to *z*.

The following syntax shows how to specify random letter and/or numeric characters in the range 0 to z in the Message Manipulations table.

Syntax:

```
Rand.string.<n>.0.z
```

where:

- **<n>** is the number of random letter and/or numeric characters you wish to specify in the range 0 to z.

The following syntax shows how to specify random numbers between *n* and *m* in the Message Manipulations table.

Syntax:

```
Rand.number.<n>.<m>
```

where:

- **<n>** specifies the start value of the range of the random numbers that you wish to specify.
- **<m>** specifies the end value of the range of the random numbers that you wish to specify.

The following table provides configuration examples for using random letters and numeric characters in the Message Manipulations table.

Table 4-1: Examples using Random Letters and Numeric Characters

Message Type	Action Subject	Action Type	Action Value
`invite.request`	`header.myrandomString`	**Add**	`Rand.string.56.A.Z`
`invite.response`	`header.NumberaAndChars`	**Add**	`Rand.string.12.0.z`
`invite.response.4xx`	`header.myrandomNmber`	**Add**	`Rand.number.50.100`

4.3 SDP Conditions

You can configure message manipulation rules based on user-defined SDP conditions.

The device supports the following SDP condition syntax:

- **Source IP Address:** You can manipulate the source IP address in the SDP. For example, you can configure a manipulation rule to add a Diversion header to incoming INVITE messages if the SDP contains a specific IP address, or a prefix or suffix of this IP address.
  - **param.message.sdp.ip suffix** '10.10'
  - **param.message.sdp.ip prefix** '10.132'
  - **param.message.sdp.ip==**'10.33.37.78'
- **RTP mode:** You can manipulate the RTP mode using the following condition:
  - **param.message.sdp.rtpmode**

  Possible values include the following:
  - sendonly
  - sendrecv
  - inactive
- **Origin IP Address:** Using the origin IP address (in the SDP "o=" line):
  - **param.message.sdp.originaddress**

  Possible values include any IP address.
- **Port:** First audio active media port number (i.e., port number greater than 0) in the "m=" field of the SDP body:
  - **sdp.port**

- **IP address:** IP address of the first active media (port greater than 0). The IP address is taken from the media "c=" field (the "c=" field below the "m=" field) of the SDP body. Note that if the "m=" field doesn't contain a "c=" field, then the IP address is taken from the global "c=" field (the "c=" field at the top of the SDP):
  - **sdp.address**

Below are manipulation examples using SDP conditions:

- **Example 1:** Copy the port and IP address in the SDP body to a customized SIP header (e.g., Custom-RTP-Address/Port) in the outgoing INVITE message, as follows:

Message Type	Action Subject	Action Type	Action Value
`invite.request`	`header.custom-rtp-address`	**`Add`**	`param.message.sdp.ip`
`invite.request`	`header.custom-rtp-port`	**`Add`**	`param.message.sdp.port`

- **Example 2:** Changes the RTP mode to sendonly if the SDP "c=" line address is 0.0.0.0:

Message Type	Condition	Action Subject	Action Type	Action Value
`reinvite.request`	`param.message.sdp.ip == '0.0.0.0'`	`param.message.sdp.rtpmode`	**`Modify`**	`'sendonly'`

- **Example 3:** Changes the SDP "c=" line to the same address as the "o=" line:

Message Type	Action Subject	Action Type	Action Value
-	`param.message.sdp.ip`	**`Modify`**	`param.message.sdp.originaddress`

- **Example 4:** Condition the RTP mode:

Message Type	Condition	Action Subject	Action Type	Action Value
`invite`	`param.message.sdp.rtpmode=='sendrecv'`	`var.call.src.1`	**`Modify`**	`'1'`
`invite.response.200`	`var.call.dst.0=='1'`	`param.message.sdp.rtpmode`	**`Modify`**	`'sendonly'`

- **Example 5:** The manipulation rule example below adds a Diversion header ("Diversion: <sip:12345@p4.isp.com>;reason=no-answer") to incoming INVITE messages if the SDP contains the IP address 10.33.37.78 or the prefix of this IP address, i.e., 10.33. The IP address is contained in the "c=" line of the SDP (e.g., "c=IN IP4 10.33.37.75"). The table below shows the example configuration:

Parameter	Rule Index 1	Rule Index 2
Message Type	`invite`	`invite`
Condition	`param.message.sdp.ip=='10.33.37.78'`	`param.message.sdp.ip prefix '10.33'`
Action Subject	`header.diversion`	`header.diversion`
Action Type	`Add`	`Add`
Action Value	`<sip:12345@p4.isp.com>;reason=no-answer`	`<sip:12345@p4.isp.com>;reason=no-answer`

You can configure several such manipulation rules and then apply them per IP Group using the 'Inbound Message Manipulation Set' parameter.

Note: This feature is applicable only to the SBC application.

4.4 Regular Expressions (Regex)

You can configure SIP header manipulation rules using regular expressions (regex). Regex is a special text string pattern matching engine which is used to define the condition that must exist in order to use a specific manipulation rule. If the SIP header matches the regex pattern, then the "action" of the manipulation rule is applied to the SIP message. Executing a regex pattern also creates sub-expressions. The sub-expressions are referenced using the *$n* syntax, where *n* is a digit in the range of 1 to 13 (e.g., $3).

Note that spaces within a regular expression must be enclosed by parenthesis, as shown in the first example below:

```
body.sdp regex (AVP 8)
body.sdp regex avp
```

This feature provides the following main benefits:

- The device does not need to know the SIP header name or structure.
- The sub-expressions can be used in the manipulation action. All that is required is to set the action (for example, add, modify, etc.) and then reference the sub-expression you want to use as the value.

This following syntax shows how to specify regular expressions (regex) in the Message Manipulations table.

Syntax:

```
<regular expression>
```

where:

- **<regular expression>** is used as part of the value in a condition and contains a regular expression.

Syntax:

```
<$n>
```

where:

- **<$n>** is used to reference a resulting sub-expression after executing a regex in a condition; where n is an integer referencing the sub-expression.

The following table provides configuration examples for using regular expressions in the Message Manipulations table.

Table 4-2: Regular Expressions Examples

Message Type	Condition	Action Subject	Action Type	Action Value
invite.request	header.diversion.0 regex (<.*)(;urlparam=[a-z]*)(.*>)	header.diversion.0	**Modify**	$1+$3
invite.request	header.diversion.0 regex (<.*)(;urlparam=[a-z]*)(.*>)	header.diversion.0	**Add**	$1 + ';mynewparam=good' + $3

Message Type	Condition	Action Subject	Action Type	Action Value
invite.response .100	header.via regex (SIP/2.0/UDP)(.*); branch=(.*)	header. thebranch	**Add**	$3
subscribe	header.to regex (.*)(1001)(.*)@(.*)>	header.to	**Modify**	$1+$3+'8@'+$4

Below are detailed examples of using regex for SIP message manipulation:

- **Example 1 - Number range matching and manipulation:**
  - Required manipulation: When the source number has prefix 30 to 40 and a digit (e.g., 3122), it needs to be changed to 2312. The last digit of the original phone number is removed (i.e., 2, leaving the number as 312) and the result is prefixed with 2.
    - Old header:

```
To: <sip:3122@10.132.10.100;user=phone
```

    - New header:

```
To: sip:2312@company244.com
```

  - Manipulation rule:

Index	Condition	Action Subject	Action Type	Action Value
1	header.to regex (<.*)(**[3-**4]**[0-**9])(.*)(\d)@(.*>)	header.to	**Modify**	$1+'2'+$2+ $3+'@'+$5

  - **Explanation:** Dialing 3122 creates the following sub-expressions:
    - 1: <sip:
    - 2: 31
    - 3: 2
    - 4: 2
    - 5: 10.132.10.100;user=phone>
- **Example 2 - Manipulation based on source and destination number:**
  - Required manipulation: If the destination number has prefix 6, 7, or 8 (e.g., 85262146) and the source number has prefix 2001, then remove the first five digits (e.g., 85262) from the destination number and add 3 as the prefix (e.g., 3146).
    - Old header:

```
From:
<sip:20011234@10.132.10.100;user=phone>;tag=XINPYDPROEOREGE
IHUHF
To: sip:85262146@10.132.10.100;user=phone
```

    - New header:

```
From: <sip:20011234@company246.com;user=phone>;tag=1c13519
To: sip:3146@company244.com
```

- Manipulation rules:

Index	Condition	Action Subject	Action Type	Action Value
1	`header.to regex <sip:([6-8][1-9]{4})(.*)@(.*>)`	`var.call.dst.0`	**Modify**	`'3'+$2`
2	`header.from regex 2001`	`header.to.url.user`	**Modify**	`var.call.dst.0`

- **Explanation:** These rules are slightly complex as both the To and From headers are inspected. This rule executes
  - If the dialed number is prefixed with a number 6-8 (inclusive)
  - If the calling party number is prefixed with 2001

  If these conditions exist, then:
  - Remove the first five digits of the dialled string.
  - Prefix the result with the digit 3.

  The first rule matches a dialled number that occurs in the To header (e.g., 85262146). If a match occurs, it uses a variable to store the remaining three digits and adds the digit 3 as the prefix. The second rule inspects the From header. If it contains the string 2001, then the user part of the To header is modified with the prepared variable. For example, the user (at 20011234) dials 85262146, which generates the following substring from the first rule:
  - $1 85262
  - $2 146
  - $3 10.132.10.100;user=phone>

Note: This configuration isolates the last three digits in the dialed number and prefixes them with '3'. The variable now is set to '3146'. The second rule does not use sub-expressions. It simply searches for 2001 in the From header and if there is a match the user part of the To header is manipulated using the standard manipulation syntax.

Example 3 - Manipulation on SDP:

- Manipulation required: To change the packet period in the SDP.
- Manipulation rule:

Index	Condition	Action Subject	Action Type	Action Value
1	`body.sdp regex (.*)(a=ptime:20)(.*)`	`body.sdp`	**Modify**	`$1+'a=ptime:10'+$3`

- **Explanation:** This rule matches everything up to the a=ptime in the SDP body as $1, and stores as $3 everything after the 0 in the ptime attribute line. This is used as the closing \r\n in the SDP body. The modify action then refers to the sub-expressions $1 and $3, but does not make use of $2, instead replacing it with a=ptime:10.

4.5 Variables for Copying Data between Messages

You can use variables in SIP message manipulation rules to copy specific information (data) from one message to another. Information from one message is copied to a variable and then information from that variable is copied to any subsequent message. The device can store information in local (call) or global variables.

To store data in a variable, add the name of the variable in the 'Action Subject' field and set the 'Action Type' to **Modify**. To retrieve data from a variable, add it in the 'Action Value' field and it can be used in any manipulation where a ManStringElement is valid as an 'Action Subject'.

4.5.1 Call Variable

The call variable stores information on a per call basis and changes when a new call is made (i.e., stored only throughout the lifetime of a specific call). Up to two local variables can be used per call: *src* (source) or *dst* (destination) references which can be stored in the call leg. Note that information stored in the call variables is only valid for the duration of the call.

Call variables use the following syntax:

```
var.call.src|dst.<0>
```

where:

- *src* denotes the call source variable.
- *dst* denotes the call destination variable
- **<0>** specifies the variable ID (note that only one source call variable can be defined).

For example:

1. Store a value in a call variable: Stores the subject URI parameter from the To header:

```
MessageManipulations 0 = 0, Invite.Request, , var.call.dst.0,
2, header.to.url.param.subject, 0;
```

2. Use the stored value: Allocates a Subject header for the 200 OK response for the same call and assigns it the stored value:

```
MessageManipulations 0 = 0, Invite.response.200, ,
header.subject, 0, var.call.dst.0, 0;
```

The following table provides additional configuration examples of using call variables in Message Manipulation rules.

Table 4-3: Examples of Call Variables

Message Type	Condition	Action Subject	Action Type	Action Value
`invite`	`param.message.sdp.rtpmode=='sendrecv'`	`var.call.src.1`	**`Modify`**	`'1'`
`invite.response.200`	`var.call.dst.0=='1'`	`param.message.sdp.rtpmode`	**`Modify`**	`'sendonly'`

4.5.2 Global Variable

Global variables are similar to call variables, but they do not change as new calls are made (i.e., their lifetime is not restricted to the duration of a call). Up to 10 global variables can be used.

Global variables use the following syntax:

```
var.global.<0-9>
```

where, **<0-9>** specifies the global variable ID.

For example:

- Store a value in a global variable: Stores the Priority header of the INVITE with 'company' in the host part of the From header:

```
MessageManipulations 0 = 0, Invite.Request,
header.from.url.host == 'company', var.global.1, 2,
header.priority, 0;
```

- Use the stored value: Assigns the same priority as the INVITE request to SUBSCRIBE requests arriving with 'company' in the host part of the From header:

```
MessageManipulations 0 = 0, Subscribe.request,
header.from.url.host == 'company', header.priority, 0,
var.global.1, 0;
```

The following table provides additional configuration examples of using variables in Message Manipulation rules.

Table 4-4: Example of Global Variables

Message Type	Condition	Action Subject	Action Type	Action Value
invite		var.global.0	**Modify**	'Custom UA'

4.5.3 Session Variable

Session variables can be preserved in any ongoing leg in the session, for example, in an call session with forking calls, in a call which had a locally handled blind transfer, etc. The value of the variable remains the same in all existing legs and in new legs of the session context.

Session variables use the following syntax:

```
var.session.0
```

where, *0* is the variable ID.

For example (using SIPRec):

For an IP-to-Tel call, the INVITE message of the recorded IP call contains the header, X-credit-card (e.g., `X-credit-card: 123456789`). When the device sends an INVITE to the SIPRec server (SRS), it is required to include the content (value) of this header (e.g., `123456789`). To do this, you need to configure two Message Manipulation rules:

1. For the recorded call: This rule stores the content of the X-credit-card header in the variable, `var.session.0`.
2. For the SRS leg: This rule adds a new header, X-credit-card with the contents of the variable (`var.session.0`) to the INVITE sent to the SRS.

Table 4-5: Example of Session Variables

Message Type	Condition	Action Subject	Action Type	Action Value	Description
`invite.request`	`header.X-credit-card exists`	`var.session.0`	**`Modify`**	`header.X-credit-card.content`	For the recorded call
`invite.request`	`var.session.0 != ''`	`header.X-credit-card`	**`Add`**	`var.session.0`	For the SRS leg

4.6 SIP Message Normalization

The device supports a built-in SIP message normalization feature that can be enabled per manipulation rule. This is enabled by setting the Action Type field to "Normalize". The normalization feature removes unknown or non-standard SIP message elements before forwarding the message. These elements can include SIP headers, SIP header parameters, and SDP body fields.

Message normalization is typically configured per SIP header but can also be configured for all headers (including SDP). For example, to normalize the Refer-To header, you would need to set the Action Subject field to "Refer-To" and the Action Type field to "Normalize".

The device normalizes the following SIP elements:

- URLs:
  - User part is normalized, for example, the bolded area is removed:
    ```
    <sip:+1-800-229-229;phone-
    context=1@10.33.2.17;user=phone;UnknownUrlParam>
    ```
  - Unknown parameters are removed, for example, the bolded area is removed:
    ```
    <sip:+1-800-229-229;phone-
    context=1@10.33.2.17;user=phone;UnknownUrlParam>
    ```

  The resultant URL after above example normalization:
  ```
  <sip:+1800229229@10.33.2.17;user=phone>
  ```
- Headers:
  - Alert-Info: unknown header parameters are removed
  - P-Called-Party-ID: unknown header parameters are removed, URL is normalized
  - P-Charging-Vector: unknown header parameters are removed
  - P-Associated-URI: unknown header parameters are removed, URL is normalized
  - P-Preferred-Identity: URL is normalized
  - Diversion: unknown header parameters are removed, URL is normalized
  - P-Asserted-Identity: URL is normalized
  - Remote-Party-ID: unknown header parameters are removed, URL is normalized
  - Reason: unknown header parameters are removed
  - Max-Forwards: value is changed to 70
  - History-Info: unknown header parameters are removed, URL is normalized
  - From: unknown header parameters are removed, URL is normalized
  - To: unknown header parameters are removed, URL is normalized
  - Via: unknown header parameters are removed
  - Refer-To: unknown header parameters are removed, URL is normalized
  - Referred-By: unknown header parameters are removed, URL is normalized
  - Event: unknown header parameters are removed
  - Session-Expires: unknown header parameters are removed
  - Min-SE: unknown header parameters are removed
  - Min-Expires: unknown header parameters are removed
  - Request-URI: URL is normalized
  - Contact: unknown header parameters are removed
  - Subscription-State: unknown header parameters are removed

For example:

- To header before normalization:

```
To: <sip:100;phone-context=1@10.33.2.17;user=phone;UnknownUrlParam>;UnknownHeaderlParam
```

- To header after SIP normalization (user parameter, unknown URL parameter, and unknown header parameter are removed):

```
To: <sip:100@10.33.2.17;user=phone>
```

- SDP Body: Removes unnecessary SDP fields (except v=, o=, s=, c=, t=, and r=) and unknown media with all its attributes. For example, the bolded text is removed before sending the message:

```
v=0
o=SMG 791285 795617 IN IP4 10.33.2.17
s=Phone-Call
i=A Seminar on the session description protocol
u=http://www.example.com/seminars/sdp.pdf
e=j.doe@example.com (Jane Doe)
c=IN IP4 10.33.2.26
t=0 0
m=unknown 6000 RTP/AVP 8
a=unknown
a=sendrecv
a=ptime:20
m=audio 6000 RTP/AVP 8
a=rtpmap:8 pcma/8000
a=sendrecv
a=unknown
a=ptime:20
```

- Message: Normalization of the entire message. Headers and bodies not listed below are removed while those listed are retained and normalized (if necessary and if listed as supported for normalization, as previously mentioned) :
  - Headers:
    - Request-URI
    - Via
    - Max-Forwards
    - From
    - To
    - Call-ID
    - Cseq
    - Contact
    - Record-Route
    - Route
    - Supported
    - Allow
    - P-Preferred-Identity
    - Diversion
    - Rack
    - Required
    - RSeq
    - Authorization

- Proxy-Authorization
- WWW-Authenticate
- Proxy-Authenticate
- Event
- Refer-To
- Referred-By
- Replaces
- User-Agent
- P-Asserted-ID
- History-Info
- Priority
- Resource-Priority
- Unsupported
- Expires
- Session-Expires
- Min-SE
- Min-Expires

- Bodies:
  - SDP
  - DTMF

Configuration Examples:

Table 4-6: Normalization Examples

Message Type	Condition	Action Subject	Action Type	Action Value	Description
`invite`	-	`message`	**`Normalize`**	-	Normalizes entire message (headers and SDP) of INVITE messages
`invite`	-	`body.sdp`	**`Normalize`**	-	Normalizes only SDP body of INVITE messages
`invite`	-	`header.max-forwards`	**`Normalize`**	-	Normalizes the Max-Forwards header of INVITE messages

This page is intentionally left blank.

5 Typical Examples

The following table provides a summary of typical examples of Message Manipulation rules.

Table 5-1: Message Manipulation Examples

Message Type	Condition	Action Subject	Action Type	Action Value	Description
invite.request	param.message.sdp.address=='flowers.com'	header.diversion	**Add**	'<sip:WeSellFlowers@p4.isp.com>;reason=time-of-day'	In INVITE requests, add a Diversion header if the c line in the SDP is set to "flowers.com".
info.response	header.request-uri.methodtype=='488'	header.request-uri.methodtype	**Modify**	'503'	Change the Request-URI method type to 503 from 403 in INFO response messages
info.response.180		header.request-uri.methodtype	**Modify**	'183'	Change request type method to 183 in 180 response messages.
invite.request	header.expires.time < '88888'	header.organisation	**Add**	'audiocodes'	Check the time parameter in Expires headers. If it is less than 88888, add an organization header to the INVITE request message.
register.request		header.contact.param.newparam	**Add**	'newValue'	Add newParam with a value of newValue as a general header level param to REGISTER Contact headers
subscribe.response		header.remote-party-id.0.partytype	**Modify**	'2'	In Subscribe response messages, change the party type to 'called' (note, 1="calling", 2= "called", 3="redirect") in the 1st Remote-Party-ID header.
invite.response		header.from.param.nasty	**Delete**		Remove the param named 'nasty' from From headers in INVITE responses.
any		header.user-agent	**Modify**	'TelcoA'	Change the User-Agent header to telcoA.
any		header.from.quote control	**Modify**	'0'	Removes quotation marks surrounding display name in From header.

This page is intentionally left blank.

A Message Manipulation Syntax Reference

This appendix provides a detailed description on the support and syntax for configuring SIP message manipulation rules.

A.1 Actions

The actions that can be done on SIP message manipulation in the Message Manipulations table are listed in the table below.

Table A-1: Message Manipulation Actions

Action	Value
Add	0
Remove	1
Modify	2
Add Prefix	3
Add Suffix	4
Remove Suffix	5
Remove Prefix	6

The maximum length of the value for a manipulation is 299 characters.

A.2 Header Types

A.2.1 Accept

An example of the header is shown below:

```
Accept:  application/sdp
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	No	N/A
Keyword	**Sub Types**			**Attributes**
N/A	N/A			N/A

Below is a header manipulation example:

Rule:	If the supported header does not contain 'mm,100rel,timer,replaces', then in all INVITE messages add an Accept header: `MessageManipulations 8 = 1, invite, header.supported != 'mm,100rel,timer,replaces', header.accept, 0, ' application/x-private ', 0;`
Result:	`Accept: application/x-private`

A.2.2 Accept-Language

An example of the header is shown below:

```
Accept-Language: da, en-gb;q=0.8, en;q=0.7
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	No	N/A
Keyword	**Sub Types**		**Attributes**	
N/A	N/A		N/A	

Below is a header manipulation example:

Rule:	Add a new Language header to all INVITE messages: `MessageManipulations 0 = 1, invite, , header.accept-language, 0, 'en, il, cz, it', 0;`
Result:	`Accept-Language: en, il, cz, it`

A.2.3 Allow

An example of the header is shown below:

```
Allow:
REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO,SUB
SCRIBE
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	No	N/A

Keyword	Sub Types	Attributes
N/A	N/A	Read/Write

Below is a header manipulation example:

Rule:	Add an Allow header to all INVITE messages: `MessageManipulations 0 = 1, invite, , header.allow, 0, 'REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO ,SUBSCRIBE, XMESSAGE', 0;`
Result:	`Allow: REGISTER,OPTIONS,INVITE,ACK,CANCEL,BYE,NOTIFY,PRACK,REFER,INFO, SUBSCRIBE, XMESSAGE`

A.2.4 Call-Id

An example of the header is shown below:

```
Call-ID: JNIYXOLCAIWTRHWOINNR@10.132.10.128
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	No	NA

Keyword	Sub Types	Attributes
ID	String	Read Only

Below is a header manipulation example:

Rule:	Add a proprietary header to all INVITE messages using the data in the Call-id header: `MessageManipulations 0 = 1, invite, , header.Xitsp-abc, 0, header.call-id, 0;`
Result:	`Xitsp-abc: GIAPOFWRBQKJVAETIODI@10.132.10.128`

A.2.5 Contact

An example of the header is shown below:

```
Contact: <sip:555@10.132.10.128:5080>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	No	8

Keyword	Sub Types	Attributes
Expires	Integer	Read/Write
GruuContact	String	Read/Write
IsGRUU	Boolean	Read/Write
Name	String	Read/Write
Param	Param	Read/Write
URL	'URL' on page 75	Read/Write*

* Host name cannot be modified in the URL structure for a contact header.

Below is a header manipulation example:

Rule:	Change the user part in the Contact header in all INVITE messages to fred: `MessageManipulations 0 = 1, Invite, ,header.contact.url.user, 2, 'fred', 0;`
Result:	`Contact: <sip:fred@10.132.10.128:5070>`

A.2.6 Cseq

An example of the header is shown below:

```
CSeq: 1 INVITE
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	No	N/A

Keyword	Sub Types	Attributes
Num	Integer	Read Only
Type	String	Read Only

Below is a header manipulation example:

Rule:	If the Cseq number is 1, then modify the user in the Contact header to fred. `MessageManipulations 0 = 1, Invite, header.cseq.num=='1',header.contact.url.user, 2, 'fred', 0;`
Result:	`Contact: <sip:fred@10.132.10.128:5070>`

A.2.7 Diversion

An example of the header is shown below:

```
Diversion: <sip:654@IPG2Host;user=phone>;reason=user-
busy;screen=no;privacy=off;counter=1
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	3

Keyword	Sub Types	Attributes
Name	String	Read/Write
Param	Param	Read/Write
Privacy	Enum Privacy (see 'Privacy' on page 79)	Read/Write
Reason	Enum Reason (see 'Reason (Diversion)' on page 79)	Read/Write
Screen	Enum Screen (see 'Screen' on page 82)	Read/Write
URL	URL Structure (see 'URL' on page 75)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Diversion header to all INVITE messages: `MessageManipulations 0 = 1, invite, , header.Diversion, 0, '<tel:+101>;reason=unknown; counter=1;screen=no; privacy=off', 0;`
	Result:	`Diversion: <tel:+101>;reason=user-busy;screen=no;privacy=off;counter=1`
Example 2	**Rule:**	Modify the Reason parameter in the header to 1, see 'Reason (Diversion)' on page 79 for possible values: `MessageManipulations 1 = 1, invite, , header.Diversion.reason, 2, '1', 0;`
	Result:	`Diversion: <tel:+101>;reason=user-busy;screen=no;privacy=off;counter=1`
Example 3	**Rule:**	The URL in the Diversion header is modified to that which is contained in the header URL: `MessageManipulations 2 = 1, invite, , header.Diversion.URL, 2, header.from.url, 0;`
	Result:	`Diversion:<sip:555@IPG2Host;user=phone>;reason=user-busy;screen=no;privacy=off;counter=1`

A.2.8 Event

An example of the header is shown below:

```
Event: foo; id=1234
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
EventKey	Event Structure (see 'Event Structure' on page 73)	Read/Write
Param	Param	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add parameter itsp-abc=voip to the Event header: `MessageManipulations 0 = 1, invite, , header.event.param.itsp-abc, 0, 'voip' , 0;`
	Result:	`Event: foo;id=1234;itsp-abc=voip`
Example 2	**Rule:**	Modify the Event ID string: `MessageManipulations 1 = 1, invite, , header.event.EVENTKEY.id, 2, '5678', 0;`
	Result:	`Event: foo;id=5678;`
Example 3	**Rule:**	Modify the Event package enum: `MessageManipulations 2 = 1, invite, , header.event.EVENTKEY.EVENTPACKAGE, 2, '2', 0;`
	Result:	`Event: refer;id=5678`

A.2.9 From

An example of the header is shown below:

```
From: <sip:555@10.132.10.128;user=phone>;tag=YQLQHCAAYBWKKRVIMWEQ
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	No	NA

Keyword	Sub Types	Attributes
Name	String	Read/Write
Param	Param	Read/Write
tag	String	Read Only
URL	URL Structure (refer to 'URL' on page 75)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Change the user part of the From header if the user is not 654: `MessageManipulations 8 = 1, invite, header.from.url.user != '654', header.from.url.user, 2, 'fred', 0;`
	Result:	`From: <sip:fred@IPG2Host;user=phone>;tag=1c20161`
Example 2	**Rule:**	Add a new parameter to the From header called p1 and set its value to myParameter: `MessageManipulations 1 = 1, Invite.request, ,header.from.param.p1, 0, 'myParameter', 0;`
	Result:	`From: <sip:fred@IPG2Host;user=phone>;p1=myParameter;tag=1c5891`
Example 3	**Rule:**	Modify the URL in the From header: `MessageManipulations 0 = 1, any, , header.from.url, 2, 'sip:3200@110.18.5.41;tusunami=0', 0;`
	Result:	`From: <sip:3200@110.18.5.41;user=phone;tusunami=0>;tag=1c23750`

A.2.10 History-Info

An example of the header is shown below:

```
History-Info: <sip:UserA@ims.example.com;index=1>
History-Info: <sip:UserA@audc.example.com;index=2>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	20

Keyword	Sub Types	Attributes
HistoryInfo	String	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a new History-Info header to the message: `MessageManipulations 0 = 1, any, , header.History-Info, 0, '<sip:UserA@audc.mydomain.com;index=3>', 0`

	Result:	`History-Info:sip:UserA@ims.example.com;index=1`
		`History-Info:sip:UserA@audc.example.com;index=2`
		`History-Info: <sip:UserA@audc.mydomain.com;index=3>`

Example 2	**Rule:**	Delete an unwanted History-Info header from the message: `MessageManipulations 0 = 1, any, , header.History-Info.1, 1, , 0;`
	Result:	`History-Info: <sip:UserA@ims.example.com;index=1>`
Example 3	**Rule:**	Delete all History-Info from the message: `MessageManipulations 0 = 1, any, , header.History-Info, 1, , 0;`
	Result:	`All history-info headers are removed.`

A.2.11 Min-Se and Min-Expires

An example of the header is shown below:

```
Min-SE: 3600
Min-Expires: 60
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Param	Param	Read/Write
Time	Integer	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Min-Se header to the message using a value of 50: `MessageManipulations 1 = 1, any, , header.min-se, 0, '50', 0;`
	Result:	`Min-SE: 50`
Example 2	**Rule:**	Modify a Min-Expires header with the min-expires value and add an additional 0: `MessageManipulations 0 = 1, Invite, , header.Min-Expires.param, 2, header.Min-Expires.time + '0', 0;`
	Result:	`Min-Expires: 340;3400`
Example 3	**Rule:**	Modify a Min-Expires header changing the time to 700: `MessageManipulations 0 = 1, Invite, , header.Min-Expires.time, 2, '700', 0;`
	Result:	`Min-Expires: 700`

A.2.12 P-Asserted-Identity

An example of the header is shown below:

```
P-Asserted-Identity: Jane Doe <sip:567@itsp.com>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	1

Keyword	Sub Types	Attributes
URL	URL Structure (see 'URL' on page 75)	Read/Write
Name	String	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a P-Asserted-Id header to all INVITE messages: `MessageManipulations 2 = 1, invite, , header.p-asserted-identity, 0, '<sip:567@itsp.com>', 0;`
	Result:	`P-Asserted-Identity: <sip:567@itsp.com>`
Example 2	**Rule:**	Modify the P-Asserted-Identity host name to be the same as the host name in the To header: `MessageManipulations 2 = 1, invite, , header.p-asserted-identity.URL.host, 2, header.to.url.host, 0;`
	Result:	`P-Asserted-Identity: <sip:567@10.132.10.128>`

A.2.13 P-Associated-Uri

An example of the header is shown below:

```
P-Associated-URI: <sip:12345678@itsp.com>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	1

Keyword	Sub Types	Attributes
Name	String	Read/Write
Param	Param	Read/Write
URL	URL Structure (see 'URL' on page 75)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a P-Associated-Uri header to all INVITE response messages: `MessageManipulations 5 = 1, register.response, ,header.P-Associated-URI, 0, '<sip:admin@10.132.10.108>', 0;`

	Result:	`P-Associated-URI:<sip:admin@10.132.10.108>`
Example 2	**Rule:**	Modify the user portion of the URL in the header to 'alice': `MessageManipulations 5 = 1, register.response, ,header.P-Associated-URI.url.user, 2, 'alice', 0;`
	Result:	`P-Associated-URI:<sip:alice@10.132.10.108>`

A.2.14 P-Called-Party-Id

An example of the header is shown below:

```
P-Called-Party-ID: <sip:2000@gw.itsp.com>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Name	String	Read/Write
URL	URL Structure (see 'URL' on page 75)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a P-Called-Party-Id header to all messages: `MessageManipulations 8 = 1, any, , header.p-called-party-id, 0, 'sip:2000@MSBG.ITSP.COM', 0;`
	Result:	`P-Called-Party-ID: <sip:2000@gw.itsp.com>`
Example 2	**Rule:**	Append a parameter (p1) to all P-Called-Party-Id headers: `MessageManipulations 9 = 1, invite, , header.p-called-party-id.param.p1, 0, 'red', 0;`
	Result:	`P-Called-Party-ID: <sip:2000@gw.itsp.com>;p1=red`
Example 3	**Rule:**	Add a display name to the P-Called-Party-Id header: `MessageManipulations 3 = 1, any, , header.p-called-party-id.name, 2, 'Secretary', 0;`
	Result:	`P-Called-Party-ID: Secretary <sip:2000@gw.itsp.com>;p1=red`

A.2.15 P-Charging-Vector

An example of the header is shown below:

```
P-Charging-Vector: icid-value=1234bc9876e; icid-generated-
at=192.0.6.8; orig-ioi=home1.net
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	No	N/A

Keyword	Sub Types	Attributes
N/A	N/A	N/A

Below are header manipulation examples:

Rule:	Add a P-Charging-Vector header to all messages: `MessageManipulations 1 = 1, any, , header.P-Charging-Vector, 0, 'icid-value=1234bc9876e; icid-generated-at=192.0.6.8; orig-ioi=home1.net', 0;`
Result:	`P-Charging-Vector: icid-value=1234bc9876e; icid-generated-at=192.0.6.8; orig-ioi=home1.net`

A.2.16 P-Preferred-Identity

An example of the header is shown below:

```
P-Preferred-Identity: "Cullen Jennings" <sip:fluffy@abc.com>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Name	String	Read/Write
URL	URL Structure (see 'URL' on page 75)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a P-Preferred-Identity header to all messages: `MessageManipulations 1 = 1, any, , header.P-Preferred-Identity, 0, 'Cullen Jennings <sip:fluffy@abc.com>', 0;`
	Result:	`P-Preferred-Identity: "Cullen Jennings" <sip:fluffy@abc.com>`
Example 2	**Rule:**	Modify the display name in the P-Preferred-Identity header: `MessageManipulations 2 = 1, any, , header.P-Preferred-Identity.name, 2, 'Alice Biloxi', 0;`
	Result:	`P-Preferred-Identity: "Alice Biloxi" <sip:fluffy@abc.com>`

A.2.17 Privacy

An example of the header is shown below:

```
Privacy: none
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	No	N/A

Keyword	Sub Types	Attributes
privacy	'Privacy Struct' on page 74	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a privacy header and set it to ''session': `MessageManipulations 1 = 1, any, , header.Privacy, 0, 'session', 0;`
	Result:	`Privacy: session`
Example 2	**Rule:**	Add 'user' to the list: `MessageManipulations 1 = 3, , , header.privacy.privacy.user, 2, '1', 0;`
	Result:	`Privacy: session;user`

A.2.18 Proxy-Require

An example of the header is shown below:

```
Proxy-Require: sec-agree
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Capabilities	SIPCapabilities Struct	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Proxy-Require header to the message: `MessageManipulations 1 = 1, any, , header.Proxy-Require, 0, 'sec-agree', 0;`
	Result:	`Proxy-Require: sec-agree`
Example 2	**Rule:**	Modify the Proxy-Require header to itsp.com: `MessageManipulations 2 = 1, any, , header.Proxy-Require, 2, 'itsp.com' , 0;`
	Result:	`Proxy-Require: itsp.com`

Example 3	**Rule:**	Set the privacy options tag in the Proxy-Require header: `MessageManipulations 0 = 0, invite, , header. Proxy-Require.privacy, 0, 1 , 0;`
	Result:	`Proxy-Require: itsp.com, privacy`

A.2.19 Reason

An example of the header is shown below:

```
Reason: SIP ;cause=200 ;text="Call completed elsewhere"
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
MLPP	MLPP Structure (see 'MLPP' on page 73)	Read/Write
Reason	Reason Structure (see 'Reason Structure' on page 74)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Reason header: `MessageManipulations 0 = 1, any, ,header.reason, 0, 'SIP;cause=200;text="Call completed elsewhere"', 0;`
	Result:	`Reason: SIP ;cause=200 ;text="Call completed elsewhere"`
Example 2	**Rule:**	Modify the reason cause number: `MessageManipulations 0 = 1, any, ,header.reason.reason.cause, 0, '200', 0;`
	Result:	`Reason: Q.850 ;cause=180 ;text="Call completed elsewhere"`
Example 3	**Rule:**	Modify the cause number: `MessageManipulations 0 = 1, any, ,header.reason.reason.reason, 0, '483', 0;`
	Result:	`Reason: SIP ;cause=483 ;text="483 Too Many Hops"`

Note: The protocol (SIP or Q.850) is controlled by setting the cause number to be greater than 0. If the cause is 0, then the text string (see Example 3) is generated from the reason number.

A.2.20 Referred-By

An example of the header is shown below:

```
Referred-By: <sip:referrer@referrer.example>;
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
param	param	Read/Write
URL	URL Structure (see 'URL' on page 75)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Referred-By header: `MessageManipulations 0 = 1, any, ,header.Referred-By, 0, '<sip:refer@refer.com>', 0;`
	Result:	`Referred-By: <sip: sip:refer@refer.com>`
Example 2	**Rule:**	Modify the host: `MessageManipulations 0 = 1, any, ,header.Referred-By.url.host, 0, 'yahoo.com', 0;`
	Result:	`Referred-By: <sip:refer@yahoo.com>`
Example 3	**Rule:**	Add a new parameter to the header: `MessageManipulations 0 = 1, any, ,header.Referred-By.param.p1, 0, 'fxs', 0`
	Result:	`Referred-By: <sip:referrer@yahoo.com>;p1=fxs`

A.2.21 Refer-To

An example of the header is shown below:

```
Refer-To: sip:conference1@example.com
Refer-To:
<sips:a8342043f@atlanta.example.com?Replaces=12345601%40atlanta.example.com%3bfrom-tag%3d314159%3bto-tag%3d1234567>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	No	N/A

Keyword	Sub Types	Attributes
N/A	N/A	N/A

Below are header manipulation examples:

Example 1	**Rule:**	Add a basic header: `MessageManipulations 0 = 1, any, ,header.Refer-to, 0, '<sip:referto@referto.com>', 0;`
	Result:	`Refer-To: <sip:referto@referto.com>`
Example 2	**Rule:**	Add a Refer-To header with URI headers: `MessageManipulations 0 = 1, any, ,header.Refer-to, 0, '<sips:a8342043f@atlanta.example.com?Replaces=12345601%40atlanta.example.com%3bfrom-tag%3d314159%3bto-tag%3d1234567>', 0;`
	Result:	`Refer-To: <sips:a8342043f@atlanta.example.com?Replaces=12345601%40atlanta.example.com%3bfrom-tag%3d314159%3bto-tag%3d1234567>`

A.2.22 Remote-Party-Id

An example of the header is shown below:

```
Remote-Party-ID: "John Smith"
<sip:john.smith@itsp.com>;party=calling; privacy=full;screen=yes
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	3

Keyword	Sub Types	Attributes
Counter	Integer	Read/Write
Name	String	Read/Write
NumberPlan	Enum Number Plan (see 'Number Plan' on page 78)	Read/Write
NumberType	Enum Number Type (see 'NumberType' on page 78)	Read/Write
Param	Param	Read/Write
Privacy	Enum Privacy (see 'Privacy' on page 79)	Read/Write
Reason	Enum Reason (RPI) (see 'Reason (Remote-Party-Id)' on page 82)	Read/Write
Screen	Enum Screen (see 'Screen' on page 82)	Read/Write
ScreenInd	Enum ScreenInd (see 'ScreenInd' on page 82)	Read/Write
URL	URL Structure (see 'URL' on page 75)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Remote-Party-Id header to the message: `MessageManipulations 0 = 1, invite, ,header.REMOTE-PARTY-ID, 0, '<sip:999@10.132.10.108>;party=calling', 0;`
	Result:	`Remote-Party-ID: <sip:999@10.132.10.108>;party=calling;npi=0;ton=0`

Example 2	**Rule:**	Create a Remote-Party-Id header using the url in the From header using the + operator to concatenate strings: `MessageManipulations 0 = 1, Invite, ,header.REMOTE-PARTY-ID, 0, '<'+header.from.url +'>' + ';party=calling', 0;`
	Result:	`Remote-Party-ID: <sip:555@10.132.10.128;user=phone>;party=calling;npi=0;ton=0`
Example 3	**Rule:**	Modify the number plan to 1 (ISDN): `MessageManipulations 1 = 1, invite, , header.Remote-Party-ID.numberplan, 2, '1', 0;`
	Result:	`Remote-Party-ID: <sip:555@10.132.10.128;user=phone>;party=calling;npi=1;ton=0`
Example 4	**Rule:**	Modify the Remote-Party-Id header to set the privacy parameter to 1 (Full): `MessageManipulations 1 = 1, invite, , header.Remote-Party-ID.privacy, 2, '1', 0;`
	Result:	`Remote-Party-ID: <sip:555@10.132.10.128;user=phone>;party=calling;privacy=full;npi=0;ton=0`

A.2.23 Request-Uri

An example of the header is shown below:

```
sip:alice:secretword@atlanta.com;transport=tcp
SIP/2.0 486 Busy Here
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	Yes	NA

Keyword	Sub Types	Attributes
Method	String	Read/Write
MethodType	Enum: ▪ 5: INVITE ▪ 6: REINVITE ▪ 7: BYE ▪ 8: OPTIONS ▪ 9: ACK ▪ 10: CANCEL ▪ 11: REGISTER ▪ 12: INFO ▪ 13: MESSAGE ▪ 14: NOTIFY ▪ 15: REFER ▪ 16: SUBSCRIBE ▪ 17: PRACK ▪ 18: UPDATE ▪ 19: PUBLISH ▪ 21: SERVICE	Read/Write
URI	String	Read/Write

Keyword	Sub Types	Attributes
URL	URL Structure (see 'URL' on page 75)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Test the Request-URI transport type. If 1 (TCP), then modify the URL portion of the From header: `MessageManipulations 1 = 1, Invite.request, header.REQUEST-URI.url.user == '101', header.REMOTE-PARTY-ID.url, 2, 'sip:3200@110.18.5.41;tusunami=0', 0;`
	Result:	`Remote-Party-ID: <sip:3200@110.18.5.41;tusunami=0>;party=calling;npi=0; ton=0`
Example 2	**Rule:**	If the method type is 5 (INVITE), then modify the Remote-Party-Id header: `MessageManipulations 2 = 1, Invite.request, header.REQUEST-URI.methodtype == '5', header.REMOTE-PARTY-ID.url, 2, 'sip:3200@110.18.5.41;tusunami=0', 0;`
	Result:	`Remote-Party-ID: <sip:3200@110.18.5.41;tusunami=0>;party=calling;npi=0; ton=0`
Example 3	**Rule:**	For all request URI's whose method types are 488, modify the message type to a 486: `MessageManipulations 1 = 1, , header.request-uri.methodtype=='488', header.request-uri.methodtype, 2, '486', 0;`
	Result:	`SIP/2.0 486 Busy Here`

A.2.24 Require

An example of the header is shown below:

```
Require: 100rel
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Capabilities	SIPCapabilities Struct	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Require header to all messages: `MessageManipulations 1 = 1, , ,header.require, 0, 'early-session,em,replaces', 0;`
	Result:	`Require: em,replaces,early-session`
Example 2	**Rule:**	If a Require header exists, then delete it: `MessageManipulations 2 = 1, Invite, header.require exists ,header.require, 1, '', 0;`
	Result:	The Require header is deleted.

Example 3	**Rule:**	Set the early media options tag in the header: `MessageManipulations 0 = 0, invite, , header.require.earlymedia, 0, 1 , 0;`
	Result:	`Require: em,replaces,early-session, early-media`
Example 4	**Rule:**	Set the privacy options tag in the Require header: `MessageManipulations 0 = 0, invite, , header.require.privacy, 0, 1 , 0;`
	Result:	`Require: em,replaces,early-session, privacy`

A.2.25 Resource-Priority

An example of the header is shown below:

```
Resource-Priority: wps.3
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	2

Keyword	Sub Types	Attributes
Namespace	String	Read/Write
RPriority	String	Read/Write

A.2.26 Retry-After

An example of the header is shown below:

```
Retry-After: 18000
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Time	Integer	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Retry-After header: `MessageManipulations 2 = 1, Invite, ,header.Retry-After, 0, '3600', 0;`
	Result:	`Retry-After: 3600`
Example 2	**Rule:**	Modify the Retry-Time in the header to 1800: `MessageManipulations 3 = 1, Invite, ,header.Retry-After.time, 2, '1800', 0;`
	Result:	`Retry-After: 1800`

A.2.27 Server or User-Agent

An example of the header is shown below:

```
User-Agent: Sip Message Generator V1.0.0.5
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
N/A	N/A	N/A

Below are header manipulation examples:

Example 1	**Rule:**	Remove the User-Agent header: `MessageManipulations 2 = 1, Invite,  ,header.user-agent, 1, '', 0;`
	Result:	The header is removed.
Example 2	**Rule:**	Change the user agent name in the header: `MessageManipulations 3 = 1, Invite,  ,header.user-agent, 2, 'itsp analogue gateway', 0;`
	Result:	`User-Agent: itsp analog gateway`

A.2.28 Service-Route

An example of the header is shown below:

```
Service-Route: <sip:P2.HOME.EXAMPLE.COM;lr>,
<sip:HSP.HOME.EXAMPLE.COM;lr>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	7

Keyword	Sub Types	Attributes
ServiceRoute	String	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add two Service-Route headers:
		`MessageManipulations 1 = 1, Invite,  ,header.service-route, 0, '<P2.HOME.EXAMPLE.COM;lr>', 0;`
		`MessageManipulations 2 = 1, Invite,  ,header.service-route, 0, '<sip:HSP.HOME.EXAMPLE.COM;lr>', 0;`

	Result:	`Service-Route:<P2.HOME.EXAMPLE.COM;lr>` `Service-Route: <sip:HSP.HOME.EXAMPLE.COM;lr>`

Example 2	**Rule:**	Modify the Service-Route header in list entry 1: `MessageManipulations 3 = 1, Invite, ,header.service-route.1.serviceroute, 2, '<sip:itsp.com;lr>', 0;`
	Result:	`Service-Route:sip:itsp.com;lr` `Service-Route: <sip:HSP.HOME.EXAMPLE.COM;lr>`
Example 3	**Rule:**	Modify the Service-Route header in list entry 0: `MessageManipulations 4 = 1, Invite, ,header.service-route.0.serviceroute, 2, '<sip:home.itsp.com;lr>', 0;`
	Result:	`Service-Route:sip:home.itsp.com;lr` `Service-Route: <sip:itsp.com;lr>`

A.2.29 Session-Expires

An example of the header is shown below:

```
Session-Expires: 480
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Param	Param	Read/Write
Refresher	Enum Refresher (see 'Refresher' on page 82)	Read/Write
Time	Integer	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add a Session-Expires header: `MessageManipulations 0 = 1, any, , header.Session-Expires, 0, '48' + '0', 0;`
	Result:	`Session-Expires: 480`
Example 2	**Rule:**	Modify the Session-Expires header to 300: `MessageManipulations 1 = 1, any, , header.Session-Expires.time, 2, '300', 0;`
	Result:	`Session-Expires: 300`
Example 3	**Rule:**	Add a param called longtimer to the header: `MessageManipulations 1 = 1, any, , header.Session-Expires.param.longtimer, 0, '5', 0;`
	Result:	`Session-Expires: 480;longtimer=5`
Example 4	**Rule:**	Set the refresher to 1 (UAC): `MessageManipulations 3 = 1, any, , header.session-expires.refresher, 2, '1', 0;`
	Result:	`Session-Expires: 300;refresher=uac;longtimer=5`

A.2.30 Subject

An example of the header is shown below:

```
Subject: A tornado is heading our way!
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Subject	String	Read/Write

Below is a header manipulation example:

Rule:	Add a Subject header: `MessageManipulations 0 = 1, any, , header.Subject, 0, 'A tornado is heading our way!', 0;`
Result:	`Subject: A tornado is heading our way!`

A.2.31 Supported

An example of the header is shown below:

```
Supported: early-session
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Capabilities	SIPCapabilities Struct	Read/Write

Below is a header manipulation example:

Example 1	**Rule:**	Add a Supported header: `MessageManipulations 1 = 1, Invite, ,header.supported, 0, 'early-session, 0;`
	Result:	`Supported: early-session`
Example 2	**Rule:**	Set path in the Supported headers options tag: `MessageManipulations 0 = 0, invite, , header.supported.path, 0, true, 0;`
	Result:	`Supported: early-session, path`

A.2.32 To

An example of the header is shown below:

```
To: <sip:101@10.132.10.128;user=phone>
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	No	NA

Keyword	Sub Types	Attributes
Name	String	Read/Write
Param	Param	Read/Write
tag	String	Read Only
URL	URL Structure (refer to 'URL' on page 75)	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Set the user phone Boolean to be false in the To header's URL: `MessageManipulations 4 = 1, invite.request, , header.to.url.UserPhone, 2, '0', 0;`
	Result:	`To: <sip:101@10.132.10.128>`
Example 2	**Rule:**	Change the URL in the To header: `MessageManipulations 4 = 1, invite.request, , header.to.url.UserPhone, 2, '0', 0;`
	Result:	`To: <sip:101@10.20.30.60:65100>`
Example 3	**Rule:**	Set the display name to 'Bob': `MessageManipulations 5 = 1, invite.request, , header.to.name, 2, 'Bob', 0;`
	Result:	`To: "Bob D" sip:101@10.20.30.60:65100`
Example 4	**Rule:**	Add a proprietary parameter to all To headers: `MessageManipulations 6 = 1, invite.request, , header.to.param.artist, 0, 'singer', 0;`
	Result:	`To: "Bob D" <sip:101@10.20.30.60:65100>;artist=singer`

A.2.33 Unsupported

An example of the header is shown below:

```
Unsupported: 100rel
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	N/A

Keyword	Sub Types	Attributes
Capabilities	SIPCapabilities Struct	Read/Write

Below are header manipulation examples:

Example 1	**Rule:**	Add an Unsupported header to the message: `MessageManipulations 0 = 1, Invite.response, ,header.unsupported, 0, 'early-session, myUnsupportedHeader', 0;`
	Result:	`Unsupported: early-session`
Example 2	**Rule:**	Modify the Unsupported header to 'replaces': `MessageManipulations 1 = 1, Invite, ,header.unsupported, 2, 'replaces', 0;`
	Result:	`Unsupported: replaces`
Example 3	**Rule:**	Set the path in the Unsupported headers options tag: `MessageManipulations 0 = 0, invite, , header.unsupported.path, 0, true, 0;`
	Result:	`Unsupported: replaces, path`

A.2.34 Via

An example of the header is shown below:

```
Via: SIP/2.0/UDP 10.132.10.128;branch=z9hG4bKUGOKMQPAVFKTAVYDQPTB
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	No	No	No	10

Keyword	Sub Types	Attributes
Alias	Boolean	Read Only
Branch	String	Read Only
Host	Host Structure (see 'Host' on page 73)	Read Only
MAddrIp	gnTIPAddress	Read Only
Param	Param	Read/Write

Keyword	Sub Types	Attributes
Port	Integer	Read Only
TransportType	Enum TransportType (see 'TransportType' on page 83)	Read Only

Below is a header manipulation example:

Rule:	Check the transport type in the first Via header and if it's set to UDP, then modify the From header's URL: `MessageManipulations 0 = 1, Invite.request, header.VIA.0.transporttype == '0', header.from.url, 2, 'sip:3200@110.18.5.41;tusunami=0', 0;`
Result:	`From: <sip:3200@110.18.5.41;user=phone;tusunami=0>;tag=1c7874`

A.2.35 Warning

An example of the header is shown below:

```
Warning: 307 isi.edu "Session parameter 'foo' not understood"
Warning: 301 isi.edu "Incompatible network address type 'E.164'"
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	1

Keyword	Sub Types	Attributes
N/A	N/A	N/A

Below is a header manipulation example:

Rule:	Add a Warning header to the message: `MessageManipulations 0 = 1, Invite.response.180, ,header.warning, 0, 'Incompatible 380', 0;`
Result:	`Warning: Incompatible 380`

A.2.36 Unknown Header

An Unknown header is a SIP header that is not included in this list of supported headers. An example of the header is shown below:

```
myExp: scooby, doo, goo, foo
```

The header properties are shown in the table below:

Header Level Action	Add	Delete	Modify	List Entries
Operations Supported	Yes	Yes	Yes	3

Keyword	Sub Types	Attributes
N/A	N/A	N/A

Below are header manipulation examples:

Example 1	**Rule:**	Add a custom header to all messages: `MessageManipulations 0 = 1, , , header.myExp, 0, 'scooby, doo, goo, foo', 0;`
	Result:	`myExp: scooby, doo, goo, foo`
Example 2	**Rule:**	Create a new header called "media", whose value is a concatenation of the time in the Session-Expires header, followed by "000", followed by ";refresher=", followed by "1" or "2", depending on whether the refresher parameter in the Session-Expires header has the value 'UAC' or 'UAS': `MessageManipulations 0 = 1, any, , header.media, 0, header.Session-Expires.time + '000' + ';refresher=' + header.Session-Expires.Refresher, 0;`
	Result:	`media: 3600000;refresher=1`

Example 3	**Rule:**	Create lists of Unknown headers:
		`MessageManipulations 1 = 1, Invite, , header.myExp.1, 0, 'scooby, doo, goo, foo1', 0;`
		`MessageManipulations 2 = 1, Invite, , header.myExp.2, 0, 'scooby, doo, goo, foo2', 0;`

	Result:	`myExp: scooby, doo, goo, foo1` `myExp: scooby, doo, goo, foo2`
Example 4	**Rule:**	Remove the SIP header 'colour' from INVITE messages: `MessageManipulations 1 = 1, Invite, , header.colour, 1, '', 0;`
	Result:	The colour header is removed.

A.3 Structure Definitions

A.3.1 Event Structure

The Event structure is used in the Event header (see 'Event' on page 53).

Table A-2: Event Structure

Keyword	Sub Types	Attributes
EventPackage	Enum Event Package (see 'Event Package' on page 77)	Read/Write
EventPackageString*	String	Read/Write
Id	String	Read/Write

Event package string is used for packages that are not listed in the Enum Event Package table (see 'Event Package' on page 77).

A.3.2 Host

The host structure is applicable to the URL structure (see 'URL' on page 75) and the Via header (see 'Via' on page 70).

Table A-3: Host Structure

Keyword	Sub Types
Port	Short
Name	String

A.3.3 MLPP

This structure is applicable to the Reason header (see 'Reason' on page 60).

Table A-4: MLPP Structure

Keyword	Sub Types
Type	Enum MLPP Reason (see 'MLPP Reason Type' on page 78)
Cause	Int

A.3.4 Privacy Struct

This structure is applicable to the Privacy header (see 'Privacy' on page 59).

Table A-5: Privacy Structure

Keyword	Sub Types
NONE	Boolean
HEADER	Boolean
SESSION	Boolean
USER	Boolean
CRITICAL	Boolean
IDENTITY	Boolean
HISTORY	Boolean

A.3.5 Reason Structure

This structure is applicable to the Reason header (see 'Reason' on page 60).

Table A-6: Reason Structure

Keyword	Sub Types
Reason	Enum Reason (see 'Reason (Reason Structure)' on page 79)
Cause	Int
Text	String

A.3.6 SIPCapabilities

This structure is applicable to the following headers:

- Supported (see 'Supported' on page 68)
- Require (see 'Require' on page 64)
- Proxy-Require (see 'Proxy-Require' on page 59)
- Unsupported (see 'Unsupported' on page 70)

Table A-7: SIPCapabilities Structure

Keyword	Sub Types
EarlyMedia	Boolean
ReliableResponse	Boolean
Timer	Boolean
EarlySession	Boolean
Privacy	Boolean
Replaces	Boolean

Keyword	Sub Types
History	Boolean
Unknown	Boolean
GRUU	Boolean
ResourcePriority	Boolean
TargetDialog	Boolean
SdpAnat	Boolean

A.3.7 URL

This structure is applicable to the following headers:

- Contact (see 'Contact' on page 51)
- Diversion (see 'Diversion' on page 52)
- From (see 'From' on page 54)
- P-Asserted-Identity (see 'P-Asserted-Identity' on page 56)
- P-Associated-Uri (see 'P-Associated-Uri' on page 56)
- P-Called-Party-Id (see 'P-Called-Party-Id' on page 57)
- P-Preferred-Identity (see 'P-Preferred-Identity' on page 58)
- Referred-By (see 'Referred-By' on page 61)
- Refer-To (see 'Refer-To' on page 61)
- Remote-Party-Id (see 'Remote-Party-Id' on page 62)
- Request-Uri (see 'Request-Uri' on page 63)
- To (see 'To' on page 69)

Table A-8: URL Structure

Keyword	Sub Types
Type	Enum Type (see 'Type' on page 83)
Host	Host Structure (see 'Host' on page 73)
MHost	Structure
UserPhone	Boolean
LooseRoute	Boolean
User	String
TransportType	Enum Transport (see 'TransportType' on page 83)
Param	Param

A.4 Random Type

Manipulation rules can include random strings and integers. An example of a manipulation rule using random values is shown below:

```
MessageManipulations 4 = 1, Invite.Request, , Header.john, 0,
rand.string.56.A.Z, 0;
```

In this example, a header called "john" is added to all INVITE messages received by the device and a random string of 56 characters containing characters A through Z is added to the header.

For a description of using random values, see the subsequent subsections.

A.4.1 Random Strings

The device can generate random strings in header manipulation rules that may be substituted where the type 'String' is required. The random string can include up to 298 characters and include a range of, for example, from a to z or 1 to 10. This string is used in the table's 'Action Value' field.

The syntax for using random strings is:

```
Rand.string.<number of characters in string>.<low character>.<high
character>
```

Examples:

- Rand.string.5.a.z: This generates a 5-character string using characters a through z.
- Rand.string.8.0.z: This generates an 8-character string using characters and digits.

A.4.2 Random Integers

The device can generate a random numeric value that may be substituted where the type 'Int' is required. The syntax for random numeric values is:

```
Rand.number.<low number>.<high number>
```

Examples:

- Rand.number.5.32: This generates an integer between 5 and 32

A.5 Enum Definitions

A.5.1 AgentRole

These ENUMs are applicable to the Server or User-Agent headers (see 'Server or User-Agent' on page 66).

Table A-9: Enum Agent Role

AgentRole	Value
Client	1
Server	2

A.5.2 Event Package

These ENUMs are applicable to the Server or User-Agent (see 'Server or User-Agent' on page 66) and Event (see 'Event' on page 53) headers.

Table A-10: Enum Event Package

Package	Value
TELEPHONY	1
REFER	2
REFRESH	3
LINE_STATUS	4
MESSAGE_SUMMARY	5
RTCPXR	6
SOFT_SYNC	7
CHECK_SYNC	8
PSTN	9
DIALOG_PACKAGE	10
REGISTRATION	11
START_CWT	12
STOP_CWT	13
UA_PROFILE	14
LINE_SEIZE	15

A.5.3 MLPP Reason Type

These ENUMs are applicable to the MLPP Structure (see 'MLPP' on page 73).

Table A-11: Enum MLPP Reason Type

Type	Value
PreEmption Reason	0
MLPP Reason	1

A.5.4 Number Plan

These ENUMs are applicable to the Remote-Party-Id header (see 'Remote-Party-Id' on page 62).

Table A-12: Enum Number Plan

Plan	Value
ISDN	1
Data	3
Telex	4
National	8
Private	9
Reserved	15

A.5.5 NumberType

These ENUMs are applicable to the Remote-Party-Id header (see 'Remote-Party-Id' on page 62).

Table A-13: Enum Number Type

Number Type	Value
INTERNATIONAL LEVEL2 REGIONAL	1
NATIONAL LEVEL1 REGIONAL	2
NETWORK PISN SPECIFIC NUMBER	3
SUBSCRIBE LOCAL	4
ABBREVIATED	6
RESERVED EXTENSION	7

A.5.6 Privacy

These ENUMs are applicable to the Remote-Party-Id (see 'Remote-Party-Id' on page 62) and Diversion (see 'Diversion' on page 52) headers.

Table A-14: Enum Privacy

Privacy Role	Value
Full	1
Off	2

A.5.7 Reason (Diversion)

These ENUMs are applicable to the Diversion header (see 'Diversion' on page 52).

Table A-15: Enum Reason

Reason	Value
Busy	1
No Answer	2
Unconditional	3
Deflection	4
Unavailable	5
No Reason	6
Out of service	7

A.5.8 Reason (Reason Structure)

These ENUMs are used in the Reason Structure (see 'Reason Structure' on page 74).

Table A-16: Enum Reason (Reason Structure)

Reason	Value
INVITE	5
REINVITE	6
BYE	7
OPTIONS	8
ACK	9
CANCEL	10
REGISTER	11
INFO	12
MESSAGE	13
NOTIFY	14

Reason	Value
REFER	15
SUBSCRIBE	16
PRACK	17
UPDATE	18
PUBLISH	19
LAST_REQUEST	20
TRYING_100	100
RINGING_180	180
CALL_FORWARD_181	181
QUEUED_182	182
SESSION_PROGRESS_183	183
OK_200	200
ACCEPTED_202	202
MULTIPLE_CHOICE_300	300
MOVED_PERMANENTLY_301	301
MOVED_TEMPORARILY_302	302
SEE_OTHER_303	303
USE_PROXY_305	305
ALTERNATIVE_SERVICE_380	380
BAD_REQUEST_400	400
UNAUTHORIZED_401	401
PAYMENT_REQUIRED_402	402
FORBIDDEN_403	403
NOT_FOUND_404	404
METHOD_NOT_ALLOWED_405	405
NOT_ACCEPTABLE_406	406
AUTHENTICATION_REQUIRED_407	407
REQUEST_TIMEOUT_408	408
CONFLICT_409	409
GONE_410	410
LENGTH_REQUIRED_411	411
CONDITIONAL_REQUEST_FAILED_412	412
REQUEST_TOO_LARGE_413	413
REQUEST_URI_TOO_LONG_414	414
UNSUPPORTED_MEDIA_415	415
UNSUPPORTED_URI_SCHEME_416	416
UNKNOWN_RESOURCE_PRIORITY_417	417

Reason	Value
BAD_EXTENSION_420	420
EXTENSION_REQUIRED_421	421
SESSION_INTERVAL_TOO_SMALL_422	422
SESSION_INTERVAL_TOO_SMALL_423	423
ANONYMITY_DISALLOWED_433	433
UNAVAILABLE_480	480
TRANSACTION_NOT_EXIST_481	481
LOOP_DETECTED_482	482
TOO_MANY_HOPS_483	483
ADDRESS_INCOMPLETE_484	484
AMBIGUOUS_485	485
BUSY_486	486
REQUEST_TERMINATED_487	
NOT_ACCEPTABLE_HERE_488	488
BAD_EVENT_489	489
REQUEST_PENDING_491	491
UNDECIPHERABLE_493	493
SECURITY_AGREEMENT_NEEDED_494	494
SERVER_INTERNAL_ERROR_500	500
NOT_IMPLEMENTED_501	501
BAD_GATEWAY_502	502
SERVICE_UNAVAILABLE_503	503
SERVER_TIME_OUT_504	504
VERSION_NOT_SUPPORTED_505	505
MESSAGE_TOO_LARGE_513	513
PRECONDITION_FAILURE_580	580
BUSY_EVERYWHERE_600	600
DECLINE_603	603
DOES_NOT_EXIST_ANYWHERE_604	604
NOT_ACCEPTABLE_606	606

A.5.9 Reason (Remote-Party-Id)

These ENUMs are applicable to the Remote-Party-Id header (see 'Remote-Party-Id' on page 62).

Table A-17: Enum Reason (RPI)

Reason	Value
Busy	1
Immediate	2
No Answer	3

A.5.10 Refresher

These ENUMs are used in the Session-Expires header (see 'Session-Expires' on page 67).

Table A-18: Enum Refresher

Refresher String	Value
UAC	1
UAS	2

A.5.11 Screen

These ENUMs are applicable to the Remote-Party-Id (see 'Remote-Party-Id' on page 62) and Diversion (see 'Diversion' on page 52) headers.

Table A-19: Enum Screen

Screen	Value
Yes	1
No	2

A.5.12 ScreenInd

These ENUMs are applicable to the Remote-Party-Id header (see 'Remote-Party-Id' on page 62).

Table A-20: Enum ScreenInd

Screen	Value
User Provided	0
User Passed	1
User Failed	2
Network Provided	3

A.5.13 TransportType

These ENUMs are applicable to the URL Structure (see 'URL' on page 75) and the Via header (see 'Via' on page 70).

Table A-21: Enum TransportType

TransportType	Value
UDP	0
TCP	1
TLS	2
SCTP	3

A.5.14 Type

These ENUMs are applicable to the URL Structure (see 'URL' on page 75).

Table A-22: Enum Type

Type	Value
SIP	1
Tel	2
Fax	3
SIPS	4

A.6 Actions and Types

Table 5-23: Action and Types

Element Type	Command Type	Command	Value Type	Remarks
IPGroup	Match	==	String	Returns true if the parameter equals to the value.
		!=	String	Returns true if the parameter not equals to the value.
		contains	String	Returns true if the string given is found in the parameter value.
		!contains	String	Returns true if the string given is not found in the parameter value.
Call-Parameter	Match	==	String	Returns true if the parameter equals to the value.
		!=	String	Returns true if the parameter not equals to the value.
		contains	String	Returns true if the string given is found in the parameter value.
		!contains	String	Returns true if the string given is not found in the parameter value.
Body	Match	==	String	Returns true if the body's content equals to the value.
		!=	String	Returns true if the body's content not equals to the value.
		contains	String	Returns true if the string given is found in the body's content.
		!contains	String	Returns true if the string given is not found in the body's content.
		exists		Returns true if this body type exists in the message.
		!exists		Returns true if this body type does not exist in the message.
	Action	Modify	String	Modifies the body content to the new value.
		Add	String	Adds a new body to the message. If such body exists the body content will be modified.
		Remove		Removes the body type from the message.
Header-List	Match	==	String *Header-list	Returns true if the header's list equals to the string.
		!=	String	Returns true if the header's list not equals to the string.

Element Type	Command Type	Command	Value Type	Remarks
			*Header-list	
		contains	String	Returns true if the header's list contains the string.
		!contains	String	Returns true if the header's list does not contain the string.
		exists		Returns true if at least one header exists in the list.
		!exists		Returns true if no headers exist in the list.
	Action	Modify	String *Header	Removes all the headers from the list and allocates a new header with the given value.
		Add	String *Header	Adds a new header to the end of the list.
		Remove		Removes the whole list from the message.
Header	Match	==	String *Header	Returns true if a header equals to the value. The header element must not be a list.
		!=	String *Header	Returns true if a header not equals to the value. The header element must not be a list.
		contains	String	Returns true if the header contains the string.
		!contains	String	Returns true if the header does not contain the string.
		exists		Returns true if the header exists.
		!exists		Returns true if the header does not exist.
	Action	Modify	String *Header	Replaces the entire header with the new value.
		Remove		Removes the header from the message, if the header is part of a list only that header will be removed.
		Add	String *Header	Adds a new header to the end of the list.
Parameter-List	Match	==	String Parameter-list	Returns true if the header's list equals to the string.
		!=	String Parameter-list	Returns true if the header's list not equals to the string.

Element Type	Command Type	Command	Value Type	Remarks
		contains	String	Returns true if the header's list contains the string.
		!contains	String	Returns true if the header's list does not contain the string.
		exists		Returns true if at least one parameter exists in the list.
		!exists		Returns true if the header's parameter list is empty.
	Action	Modify	String Parameter-list	Replaces the current parameters with the new value.
		Add	String Parameter	Adds a new parameter to the parameter's list.
		Remove		Removes all the unknown parameters from the list.
Parameter	Match	==	String Parameter	Returns true if the header's parameter's value equals to the value.
		!=	String Parameter	Returns true if the header's parameter's value not equals to the value.
		contains	String	Returns true if the header's parameter contains the string.
		!contains	String	Returns true if the header's parameter does not contain the string.
		exists		Returns true if the header's parameter exists.
		!exists		Returns true if the header's parameter does not exist.
	Action	Modify	String Parameter	Sets the header's parameter to the value.
		Remove		Removes the header's parameter from the parameter list.
Structure	Match	==	String *Structure	Returns true if the header's structure's value equals to the value. The string given must be able to be parsed to the structure.
		!=	String *Structure	Returns true if the header's structure's value not equals to the value. The string given must be able to be parsed to the structure.
	Action	Modify	String *Structure	Sets the header's structure to the value. The string given must be able to be parsed to the structure.

Element Type	Command Type	Command	Value Type	Remarks
Integer	Match	==	Integer	Returns true if value equals to the integer element
		!=	Integer	Returns true if value not equals to the integer element
		>	Integer	Returns true if value is greater than the value.
		>=	Integer	Returns true if value is greater than or equals to the value.
		<	Integer	Returns true if value is less than the value.
		<=	Integer	Returns true if value is less than or equals to the value.
	Action	Modify	Integer	Sets the integer element to the value. A string value must be a representation of an integer.
String	Match	==	String	Returns true if the string element equals to the value.
		!=	String	Returns true if the string element not equals to the value.
		contains	String	Returns true if the value is found in the string element.
		!contains	String	Returns true if the value is not found in the string element.
		>	String	Performs a character by character compare. Returns true if the ASCII value of the character is greater than that in the value
		>=	String	Performs a character by character compare. Returns true if the ASCII value of the character is greater than or equal to that in the value
		<	String	Performs a character by character compare. Returns true if the ASCII value of the character is less than that in the value
		<=	String	Performs a character by character compare. Returns true if the ASCII value of the character is less than or equal to that in the value
	Action	Modify	String	Sets the string element to the value.
		Add prefix	String	Adds the value to the beginning of the string element.
		Remove prefix	String	Removes the value from the beginning of the string element.
		Add suffix	String	Adds the value to the end of the string element.
		Remove suffix	String	Removes the value from the end of the string element.

Element Type	Command Type	Command	Value Type	Remarks
Boolean	Match	==	Boolean	Returns true if the Boolean element equals to the value. Boolean – can be either 0 or 1.
		!=	Boolean	Returns true if the Boolean element not equals to the value. Boolean – can be either 0 or 1.
		>	Boolean	Returns true if the Boolean element not equals to the value. Boolean – can be either 0 or 1.
		<	Boolean	Returns true if the Boolean element not equals to the value. Boolean – can be either 0 or 1.
	Action	Modify	Boolean	Sets the Boolean element to the value. Boolean – can be either 0 or 1.
Attribute	Match	==	Integer *Attribute	Returns true if the attribute element equals to the value. An attribute element value must be of the same type of the attribute element.
		!=	Integer *Attribute	Returns true if the attribute element not equals to the value. An attribute element value must be of the same type of the attribute element.
	Action	Modify	Integer *Attribute	Sets the attribute element to the value. An attribute element value must be of the same type of the attribute element.

A.7 Syntax

This section describes the fields of the Message manipulations table:

Man Set ID	Message Type	Condition	Action Subject	Action Type	Action Value	Row Rule

A.8 Message Type

Description: Rule is applied only if this is the message's type
Syntax: <method>.<message role>

- **Method:**
  - **Description:** Rule is applied only if this is the message's method
  - **Syntax:** token / any
  - **Examples:**
    - invite, subscribe — rule applies only to INVITE messages
    - unknown — unknown methods are also allowed
    - any — no limitation on the method type
- **Message role:**
  - **Description:** Rule is applied only if this is the message's role
  - **Syntax:** request / response.response-code / any
  - **Examples:**
    - request — rule applies only on requests
    - response.200 — rule applies only on 200 OK messages
    - any — no limitations on the type of the message
- **Response code:**
  - **Description:** Response code of the message
  - **Syntax:** 1xx / 2xx / 3xx / 4xx / 5xx / 6xx / 3digit / any
  - **Examples:**
    - 3xx — any redirection response
    - 200 — only 200 OK response
    - Any — any response

Examples:

- invite.request
- invite.response.200
- subscribe.response.2xx

A.9 Condition

Description: Matching criteria for the rule

Syntax: (Action Subject / param) SWS match-type [SWS Action Value] * [SWS logical-expression SWS Condition]

Examples:

- header.from.user == '100'
- header.contact.header-param.expires > '3600'
- header.to.host contains 'itsp'
- param.call.dst.user != '100'
- header.john exists
- header.john exists AND header.to.host !contains 'john'
- header.from.user == '100' OR header.from.user == '102' OR header.from.user == '300'
- **match-type**
  - **Description:** Comparison to be made
  - **Syntax:**
    - == equals
    - != not equals
    - > greater than
    - < less than
    - >= greater than or equal to
    - <= less than or equal to
    - contains does a string contain a value (relevant only to string fields)
    - exists does a certain header exists
    - !exists does a certain header not exists
    - !contains does a string exclude a value. Relevant only to string fields
- **logical-expression:**
  - **Description:** Condition for the logical expression
  - **Syntax:**
    - AND logical And
    - OR logical Or

  Note: "A AND B OR C" is calculated as A AND (B OR C).

A.10 Action Subject

Description: Element in the message

Syntax: (header / body).Action Subject name [.header-index] * [.(sub-element / sub-element-param)]

Examples:

- header.from
- header.via.2.host
- header.contact.header-param.expires
- header.to.uri-param.user-param
- body.application/dtmf-relay

- **Action Subject name:**
  - **Description:** Name of the message's element - "/" only used for body types
  - **Syntax:** 1 * (token / "/")
  - **Examples:**
    - from (header's name)
    - to (header's name)
    - application/dtmf-relay (body's name)
- **header-index:**
  - **Description:** Header's index in the list of headers
  - **Syntax:** Integer
  - **Examples:** If five Via headers arrive:
    - 0 (default) refers to first Via header in message
    - 1 second Via header
    - 4 fifth Via header
- **sub-element:**
  - **Description:** Header's element
  - **Syntax:** sub-element-name
  - **Examples:**
    - user
    - host
- **sub-element-param:**
  - **Description:** Header's element
  - **Syntax:** sub-element-name [.sub-element-param-name]
  - **Example:**
    - header.from.param.expires
- **sub-element-param-name**
  - **Description:** Header's parameter name - relevant only to parameter sub-elements
  - **Syntax:** token
  - **Examples:**
    - expires (contact's header's param)
    - duration (retry-after header's param)
    - unknown-param (any unknown param can be added/removed from the header)

- **param:**
  - **Description:** Params can be as values for match and action
  - **Syntax:** param.param-sub-element.param-dir-element.(call-param-entity / ipg-param-entity)
  - **Examples:**
    - param.ipg. src.user
    - param.ipg.dst.host
    - param.ipg.src.type
    - param.call.src.user
- **param-sub-element:**
  - **Description:** Determines whether the param being accessed is a call or an IP Group
  - **Syntax:**
    - call relates to source or destination URI for the call
    - ipg relates to source or destination IP Group
- **param-dir-element:**
  - **Description:** Direction relating to the classification
  - **Syntax:**
    - src refers to source
    - ds refers to destination
- **call-param-entity**
  - **Description:** Parameters that can be accessed on the call
  - **Syntax:**
    - user refers to username in request-URI for call
- **ipg-param-entity:**
  - **Description:** Name of the parameter
  - **Syntax:**
    - user refers to Contact user in IP Group
    - host refers to Group Name in IP Group table
    - type refers to Type field in IP Group table
    - id refers to IP Group ID (used to identify source or destination IP Group)
- **string:**
  - **Description:** String
  - **Syntax:** string enclosed in single apostrophe
  - **Examples:**
    - 'username
    - '123'
    - 'user@host'
- **Integer:**
  - **Description:** A number
  - **Syntax:** 1 * digit
  - **Example:**
    - 123

A.11 Action Type

Description: Action to be performed on the element

Syntax:

- modify sets element to new value (all element types)
- add-prefix adds value at beginning of string (string element only)
- remove-prefix removes value from beginning of string (string element only)
- add-suffix adds value at end of string (string element only)
- remove-suffix removes value from end of string (string element only)
- add adds a new header/param/body (header or parameter elements)
- remove removes a header/param/body (header or parameter elements)

A.12 Action Value

Description: Value for action and match

Syntax: ('string' / Action Subject / param) * (+ ('string' / Action Subject / param))

Examples:

- 'itsp.com'
- header.from.user
- param.ipg.src.user
- param.ipg.dst.host + '.com'
- param.call.src.user + '<' + header.from.user + '@' + header.p-asserted-id.host + '>'

International Headquarters
1 Hayarden Street,
Airport City
Lod 7019900, Israel
Tel: +972-3-976-4000
Fax: +972-3-976-4040

AudioCodes Inc.
27 World's Fair Drive,
Somerset, NJ 08873
Tel: +1-732-469-0880
Fax: +1-732-469-2298

Contact us: www.audiocodes.com/info
Website: www.audiocodes.com

Document #: LTRT-28629