



IxLoad Voice Test Library Reference

Release 8.40EA-Update1

Notices

Copyright Notice

© Keysight Technologies 2004–2018

No part of this document may be reproduced in any form or by any means (including electronic storage and retrieval or translation into a foreign language) without prior agreement and written consent from Keysight Technologies, Inc. as governed by United States and international copyright laws.

Warranty

The material contained in this document is provided “as is,” and is subject to being changed, without notice, in future editions. Further, to the maximum extent permitted by applicable law, Keysight disclaims all warranties, either express or implied, with regard to this manual and any information contained herein, including but not limited to the implied warranties of merchantability and fitness for a particular purpose. Keysight shall not be liable for errors or for incidental or consequential damages in connection with the furnishing, use, or performance of this document or of any information contained herein. Should Keysight and the user have a separate written agreement with warranty terms covering the material in this document that conflict with these terms, the warranty terms in the separate agreement shall control.

Technology Licenses

The hardware and/or software described in this document are furnished under a license and may be used or copied only in accordance with the terms of such license.

U.S. Government Rights

The Software is “commercial computer software,” as defined by Federal Acquisition Regulation (“FAR”) 2.101. Pursuant to FAR 12.212 and 27.405-3 and Department of Defense FAR Supplement (“DFARS”) 227.7202, the U.S. government

acquires commercial computer software under the same terms by which the software is customarily provided to the public. Accordingly, Keysight provides the Software to U.S. government customers under its standard commercial license, which is embodied in its End User License Agreement (EULA), a copy of which can be found at

<http://www.keysight.com/find/sweula> or <https://support.ixiacom.com/support-services/warranty-license-agreements>.

The license set forth in the EULA represents the exclusive authority by which the U.S. government may use, modify, distribute, or disclose the Software. The EULA and the license set forth therein, does not require or permit, among other things, that Keysight: (1) Furnish technical information related to commercial computer software or commercial computer software documentation that is not customarily provided to the public; or (2) Relinquish to, or otherwise provide, the government rights in excess of these rights customarily provided to the public to use, modify, reproduce, release, perform, display, or disclose commercial computer software or commercial computer software documentation. No additional government requirements beyond those set forth in the EULA shall apply, except to the extent that those terms, rights, or licenses are explicitly required from all providers of commercial computer software pursuant to the FAR and the DFARS and are set forth specifically in writing elsewhere in the EULA. Keysight shall be under no obligation to update, revise or otherwise modify the Software. With respect to any technical data as defined by FAR 2.101, pursuant to FAR 12.211 and 27.404.2 and DFARS 227.7102, the U.S. government acquires no greater than Limited Rights as defined in FAR 27.401 or DFAR 227.7103-5 (c), as applicable in any technical data. 52.227-14 (June 1987) or DFAR 252.227-7015 (b) (2) (November 1995), as applicable in any technical data.

Contact Us

Ixia headquarters

26601 West Agoura Road
Calabasas, California 91302
+1 877 367 4942 – Toll-free North America
+1 818 871 1800 – Outside North America
+1.818.871.1805 – Fax
www.ixiacom.com/contact/info

Support

Global Support	+1 818 595 2599	support@ixiacom.com
APAC Support	+91 80 4939 6410	support-asiapac@ixiacom.com
EMEA Support	+40 21 301 5699	support-emea@ixiacom.com
Greater China Region	+400 898 0598	support-china@ixiacom.com
Hong Kong	+86 10 5732 3932	support-china@ixiacom.com
India Office	+91 80 4939 6410	support-india@ixiacom.com
Japan Head Office	+81 3 5326 1980	support-japan@ixiacom.com
Korea Office	+82 2 3461 0095	support-korea@ixiacom.com
Singapore Office	+656 494 8910	support-asiapac@ixiacom.com

CONTENTS

Contact Us	iii
About this guide	1
Note icons and messages	2
Typographical conventions	2
Textual conventions	3
Chapter 1 Introduction to the Voice Test Libraries	5
Who Should Read This Manual	6
Voice Test Libraries Overview	7
Voice Libraries Function Sets	8
Chapter 2 Voice Test Libraries Settings	15
Global Settings	16
Chapter 3 Voice Functions Reference	31
VoIP SIP Functions Library	32
VoIP Skinny Functions Library	65
VoIP Media Functions Library	95
VoIP Flow Functions Library	119
VoIP H323 RAS Library	124
VoIP H248 Functions Library	127
VoIP HTTP Functions Library	139
VoIP MGCP Functions Library	145
Digital T1/E1 Functions Library	160
Chapter 4 Basic Test Scenarios and Procedures	177

SIP Procedures, Sample Test Configurations, and Test Scenarios	178
Skinny Procedures, Sample Test Configurations, and Test Scenarios	252
H.323 Sample Test Configuratins and Test Scenarios	322
H.248 Sample Test Configuratins and Test Scenarios	332
MGCP Sample Test Configuratins and Test Scenarios	360
PSTN Sample Test Configuratins and Test Scenarios	377
Chapter 5 Extended Functionality SIP Tests Suite	383
SIP Tests Suite Overview	384
Predefined Common Test Procedures	388
Test Descriptions	391
Appendix A Creating a SIP Message from Template	417
SIP Message Elements	418
Appendix B The Expression Evaluator Syntax	423
Data Types	424
Operators	425
Appendix C Using the H248 Descriptive Editor	431
The Descriptive Editor GUI	432
Appendix D Using the MGCP Parameter Editor	439
MGCP Script Functions Overview	440
Appendix E Skinny Sample Configurations Overview	445
Appendix F Support for Multipart SIP Messages	453

About this guide




This section explains the notational and typographical conventions used in this documentation, and provides a list of related documentation.

Section contents:

- Note icons and messages** 2
- Typographical conventions** 2
- Textual conventions** 3

Note icons and messages

The following table describes the note icons and messages used in this document.

Name	Icon	Description
Note		Indicates information that emphasizes or supplements important points in the main text.
Important		Indicates information that is essential to the completion of a task.
Tip		Provides supplemental suggestions for applying techniques and procedures to accomplish a task.

Typographical conventions

The following table describes the typographical conventions used in this document.

Convention	Description
bold text	Bold text indicates: <ul style="list-style-type: none">• Graphical user interface element names (such as dialog boxes, buttons, menu selections, and so forth).• Command line interface commands and options.• The name of a field, option, or parameter when used as part of an instruction. For example: "Select the desired Inner Priority value."
<i>Italic text</i>	Italic text indicates: <ul style="list-style-type: none">• A text reference to the name of a field, option, or parameter.• Document and book titles.• The first reference to a new term.• Special identification or emphasis in a statement.
monospace text	Text shown in a monospace font is used to indicate: <ul style="list-style-type: none">• Text that you input. For example: Enter <code>cd project1</code>• Code samples, including keywords and variables within text and as separate paragraphs, and user-defined program elements within text.• Text file content or examples, such as lines of text from an .ini file.

Textual conventions

The following table describes the typographical conventions used in this document.

Convention	Description
Boolean text	Words that represent Boolean notation are written in all uppercase text. For example: 192.168.5.5 OR 10.20.30.30.
Keystrokes	Simultaneous keystrokes are shown by joining the key names with a plus sign (+). For example, CTRL+Q.
> (right angle bracket)	Separates levels in a hierarchy of menu selections. For example: "Select Administration > Users. "

This page intentionally left blank.

CHAPTER 1 Introduction to the Voice Test Libraries

In this chapter:

- Who Should Read This Manual 6**
- Voice Test Libraries Overview 7**
- Voice Libraries Function Sets 8**

Who Should Read This Manual

This manual is intended for users intending to edit VoIP and PSTN tests using the Voice Test Libraries of the IxLoad Voice Plug-In application.

This manual describes the Voice Test Libraries, their sets of script functions, their associated global and local parameters. The manual also includes a description of the predefined Voice scenario variables that can be used within Voice test scenarios.

Voice Test Libraries Overview

The VoIP SIP Test Library includes test functions for establishing calls using the SIP signaling protocol, and for generating RTP media streaming across established calls.

The VoIP Test Skinny Library functions assemble fully featured SCCP clients by simulating Cisco SKINNY phones that generate and receive Skinny messages, and generate RTP media streaming across calls.

In addition to SIP, Skinny, and RTP functions, test flows use special call control functions from the Flow Test Library that indicate the start/end of an execution flow, test for variable status at a particular moment in the flow, and so on. Flow functions are fully compatible with the functions in the SIP, Skinny, and RTP libraries and can be used conjointly with these functions in building test scenarios.

The functions in the Voice test libraries are grouped by protocols as follows:

- [VoIP SIP Functions Library](#)
- [VoIP Skinny Functions Library](#)
- [VoIP Media Functions Library](#)
- [VoIP Flow Functions Library](#)
- [VoIP H323 RAS Library](#)
- [VoIP H323 Functions Library](#)
- [VoIP H248 Functions Library](#)
- [VoIP MGCP Functions Library](#)
- [Digital T1/E1 Functions Library](#)

Voice Libraries Function Sets

The tables in this section describe the script functions currently implemented in the Voice test libraries.

SIP Library Functions Set

The following table briefly describes the functions implemented in the VoIP SIP test library:

Function	Description
Send Request	Sends a SIP request.
Send Response	Sends a SIP response.
Wait Request	Waits for a SIP request.
Wait Response	Waits for a SIP response.
Wait Message	Waits for a SIP message and matches it with one of the message templates defined within the function.
Retransmit Last Message	Re-sends the last transmitted SIP message.
Extract Variables	Extracts scenario variables from SIP messages.

Skinny Library Functions Set

The following table briefly describes the functions implemented in the VoIP Skinny test library:

Function	Description
OffHook	Notifies the Cisco CallManager that a terminal is in an offhook condition.
OnHook	Notifies the Cisco CallManager that a station is now in an on-hook condition.
NewCall	Sends a SoftKeyEvent message to the Cisco CallManager, requesting dial tone.
EndCall	Sends a SoftKeyEvent message to the Cisco CallManager, requesting a specific call completion.
MakeCall	Originates a call by dialing the phone number and performing the call establishment.
WaitCall	Waits for an incoming call.
AnswerCall	Answers an incoming call by going off-hook and performing the call establishment.

DialDigits	Dials the specified digits.
WaitDigits	Waits for a specified digits sequence.
HoldCall	Performs a hold operation on a specified call reference.
RetrieveCall	Performs a retrieve operation on a specified call reference.
SetupXfer	Initiates a transfer or a conference procedure.
CompleteXfer	Completes a transfer or a conference procedure.
Transfer	As a combination of the SetupXfer and CompleteXfer functions, it sets up a transfer or a conference.
ForwardAllCalls	Sends a SoftKeyEvent message to the Cisco CallManager, requesting that all incoming calls be forwarded.
ParkCall	Parks a specific call.
RegisterClient	Registers the Skinny client with a Cisco CallManager.
UnregisterClient	Unregisters the Skinny client from a Cisco CallManager.
GetCallInfo	Retrieves the call information into the predefined VoIP Skinny variables.
MeetMe	Sets up a MeetMe type conference call.
RemoveLast ConferenceParty	Removes from a conference call the party that has joined last.
SendStimulus	A Skinny Client uses this message to inform the Cisco CallManager that a functional stimulus button was clicked.
SendSoftKey	Emulated stations of the CP-7940/60 type use this message to inform the Cisco CallManager of a softkey event.
IsSoftKeyAvailable	Verifies if a certain soft key is available.
WaitForEvent	Is used by a Skinny Client to search for a specified message in the message queue. If the message does not exist, the function waits for the specified message.

RTP Library Functions Set

The following table briefly describes the functions implemented in the VoIP RTP test library:

Function	Description
Generate DTMF	Generates a specified DTMF sequence.

Detect DTMF	Detects a sequence of DTMF signals.
Generate MF	Generates a specified MF sequence.
Detect MF	Detects a sequence of MF signals.
Generate Tone	Generates a custom tone.
Wait For Tone	Detects a custom tone.
RTP Control	Checks for RTP completion or terminates a RTP script function.
Path Confirmation	Executes a Path Confirmation sequence using DTMFs, MFs, or Custom Tones.
Talk	Plays the specified wave files across the established call.
Listen	Listens to the specified wave files across the established call.
Voice Session	Plays and records simultaneously the specified wave files across the established call.
Multimedia Session	Plays an audio file and a video MP4 file across an established SIP or H.323 call.
T38 Fax Session	Sends or receives a specified image file across a fax session negotiated using the SIP signaling protocol.

Flow Functions Set

The following table lists the functions implemented in the Voice Flow test library:

Function	Description
Start	Indicates the beginning of a Test Scenario flow on the channel.
Stop	Indicates the end of an execution thread or the end of the entire Test Scenario.
Variable Set	Declares and sets the variable values to use in the test scenario.
Variable Test	Assesses a series of logical variables.
Sleep	Applies a static or random delay to the execution flow.
Procedure	Declares a procedure in the current test scenario.
Exit Proc	Determines the output of a procedure.
Counter Op	Inserts user-defined statistic counters, which may be incremented, decremented, or reset.

Test Time	Assesses the current time.
Log Message	Enables you to edit messages to include in the execution log.
Dump Variable	Generates a log containing all variables from all engines and the user-defined variables.
Error Handler	This script function can be used to minimize the number of connectors in a script.

MGCP Functions Set

The following table lists the functions implemented in the VoIP MGCP test library:

Function	Description
Send NOTIFY	The Send Notify script function implements the transaction initiated by a Notify command.
Send DLCX (GW)	The Send DLCX script function implements the transaction initiated by a sent DeleteConnection command.
Send RSIP	The Send RSIP script function implements the transaction initiated by a RestartInProgress (RSIP) command sent from the MGW to the CA.
Wait CRCX	The Wait CRCX script function implements the transaction initiated by the receiving of a CreateConnection (CRCX) command, followed by the sending of a response to this command.
Wait DLCX	The Wait DLCX script function implements the transaction initiated by the receiving of a DeleteConnection (CRCX) command, followed by the sending of a response to this command.
Wait MDCX	The Wait MDCX script function implements the transaction initiated by a the receiving of a ModifyConnection (CRCX) command, followed by the sending of a response to this command.
Wait RQNT	The Wait RQNT script function implements the transaction initiated by the receiving of a RequestNotify (RQNT) command, followed by the sending of a response to this command.
Wait AUEP	The Wait AUEP script function implements the transaction initiated by the receiving of an AuditEndpoint (AUEP) command, followed by the sending of a response to this command.
Wait AUCX	The Wait AUCX script function implements the transaction initiated by the receiving of an AuditConnection (AUCX) command, followed by the sending of a response to this command.

Wait EPCF	The Wait EPCF script function implements the transaction initiated by the receiving of an EndpointConfiguration (EPCF) command, followed by the sending of a response to this command.
Wait Command (GW)	The Wait Command (GW) function specifies one or more MGCP commands awaited by the MGW. If any of these commands is received, the user-configured response to the command is sent and, if executed successfully, the function exits on the output identifying the matched command.
Send RQNT	The Send RQNT script function implements the transaction initiated by a RequestNotify (RQNT) command sent by the CA.
Send CRCX	The Send CRCX script function implements the transaction initiated by a CreateConnection (CRCX) command sent by the CA.
Send DLCX	The Send DLCX script function implements the transaction initiated by a DeleteConnection (DLCX) command sent by the CA.
Send MDCX	The Send MDCX script function implements the transaction initiated by a ModifyConnection (MDCX) command sent by the CA.
Send AUCX	The Send AUCX script function implements the transaction initiated by an AuditConnection (AUCX) command sent by the CA.
Send AUEP	The Send AUEP script function implements the transaction initiated by a Audit Connection (AUEP) command sent by the CA.
Send EPCF	The Send EPCF script function implements the transaction initiated by a Endpoint Configuration (EPCF) command sent by the CA.
Wait NTFY	The Wait NTFY script function implements the transaction initiated by the receiving of an MGCP Notify command sent by an MGW.
Wait DLCX	The Wait DLCX script function implements the transaction initiated by the receiving at the CA of an MGCP DeleteConnection (DCLX) command sent by an MGW.
Wait Command (CA)	The Wait Command (CA) script function specifies one or more MGCP commands awaited by the CA. If any of these commands is received, the user-configured response to the command is sent and, if executed successfully, the function exits on the output identifying the matched command.
Wait RSIP	The Wait RSIP script function implements the transaction initiated by the receiving at the CA of an MGCP RestartInProgress (RSIP) command sent by an MGW.

T1/E1 Functions Set

The following table lists the functions implemented in the Digital T1/E1 test library:

Function	Description
----------	-------------

Make Call	This script function initiates a call to the specified destination.
Receive Call	This script function answers an incoming call.
End Call	This script function terminates an established call.
Path Confirmation	This script function executes a Path Confirmation sequence, wherein the path confirmation initiator sends a specific digit (DTMF/MF/tone) sequence and then waits to receive another digit (DTMF/MF/Tone) sequence from the remote party.
Talk	This script function plays back a wave file from the IxLoad Wave Files pool.
Listen	This script function allows the recording of a wave file for a specified duration.
Voice Session	This script function plays back a wave file and records a wave file at the same time.
Generate DTMF	This script function generates a sequence of Dual Tone Multiple Frequency (DTMF) signals.
Detect DTMF	This script function is used to detect a sequence of DTMF signals.
Generate MF	This script function generates a sequence of Multi-Frequency (MF) tones.
Detect MF	This script function is used to detect a sequence of MF tones.
Generate Tone	This script function generates a single custom tone (single or dual continuous, or cadence) that can be selected from the Custom Tones Pool.
Wait for Tone	This script function detects any custom tone from a user configured tone list.

This page intentionally left blank.


CHAPTER 2 Voice Test Libraries Settings

In this chapter:

Global Settings 16

Global Settings


The Global Settings represent application-wide, default test execution parameters associated with the IxLoad Voice Plug-in environment and the test libraries supported by it. Configured Global Settings are taken into account, for example, when placing a function in the Scenario Editor, when running a test execution session and when using the application workspace.

The Global Settings window is accessed by clicking the  button in the Scenario Editor window. The following Global Settings categories represent Voice test library settings:

- **Library Settings and Outputs:** This category represents the global settings for the most frequently used parameters of the functions in the Voice library. A Voice script function that is initially placed in the Scenario Editor can be configured to use a subset of these library settings.
- **Scenario Editor Defaults:** This category specifies the default settings for each SIP, Skinny, H.323, MGCP, H.248/MEGACO, Digital T1/E1, RTP, and Flow script function that has been placed in the Scenario Editor.

Configuring Library Settings and Outputs


Click **Library Settings and Outputs** in the left pane of the **Global Settings** window to configure a subset of the VoIP and T1/E1 test library parameters. This settings category also enables you to define the default output names for Voice library functions.

 **Note:** These settings are used by a VoIP or T1/E1 script function placed in the Scenario Editor only if the function's **Use Global Settings** parameter is selected.

RTP Settings

The following table lists the global VoIP RTP library settings:






Name	Description
RTP Transmission Mode Specifies the RTP transmission mode, which can be one of the following: <ul style="list-style-type: none"> • In Band (using RTP media streaming) • Out of Band – Using 2833 EVENT Payload Format • Out of Band – Using 2833 TONE Payload Format 	
RTP Playback	
Play x time(s)	Sets the number of times the wave file is played.
Repeat Continuous for	Sets the time period the wave file is played (in seconds, minutes, or hours).
Use Talk Time (only for BHCA/CPS objective)	In the case of a test configured with a BHCA or CPS objective, choosing this option plays back the wave file for the duration of the talk time parameter.
Audio Playback	
Output Volume	Sets the volume level during the VoIP communication. The available values include: <ul style="list-style-type: none"> • -15 dB (default) • -20 dB • -25 dB • -30 dB
Path Confirmation: Specifies the DTMF/MF/Tone generation and detection settings for the RTP Path Confirmation function.	
DTMF/MF/TONE Generation	

Tone Duration	The time duration (in ms) of a single tone. The range of values is 40 to 59960 ms. The default value is 200 ms.
Inter Tone Interval	The maximum amount of time (in ms) between two consecutive generated DTMFs. The range of values is 40 to 59960 ms. The default value is 200 ms.  Note: The sum of the tone duration and the inter tone interval is required to be less than 60000 ms.
Tone Amplitude	The attenuation (in dB) of the DTMF tone. The minimum attenuation is 0 dB (no attenuation) and the maximum is -40 dB. The default value is -10 dB.
DTMF/MF/TONE Detection	
First Sequence Timeout	The time (in ms) allowed for receiving the first digit. After this period elapses, the function exits on Timeout output. The range of values is 200 to 999999 ms. The default value is 5000 ms.
Inter sequence Timeout	The maximum amount of time (in ms) allowed between consecutive DTMFs/MFs/custom tones sequences for a proper detection. After this period elapses, the function exits on Timeout output. The range of values is 200 to 999999 ms. The default value is 5000 ms.
QoV - When performing QoV computations, adding silence periods and increasing the list duration enables the application to correctly resolve loops and identify the played clip.	
Talk - Add extra silence	If selected, this adds to the Talk function a silence period of the duration specified in the adjacent field.
Listen - Increase duration by	If selected, this adds to the Listen function a silence period of the duration specified in the adjacent field.
Path Confirmation Sequence	
Execute once	The path confirmation sequence is executed once.
Execute for	The path confirmation sequence is executed for a specified time interval.

Skinny Settings


The following table lists the global Skinny library settings:

Name	Description
------	-------------

SoftKeys	<p>The editable table displays an available softkeys list. The following operations can be performed on the softkeys table:</p> <ul style="list-style-type: none">  Adds a softkey table entry.  Deletes a softkey table entry.  Displays a softkey's properties.  Imports softkeys from a user-specified file.  Exports softkeys to a user-specified file.
----------	---

SIP Settings

The following table lists the global SIP library settings:

Name	Description
Transport Layer	
Disable TCP Support	<p>If this checkbox is enabled, support for the TCP protocol is disabled. The default value is Disabled.</p> <hr/> <p> Note: You need to disable TCP support, for example, when performing SIP performance testing on UDP transport.</p>
Authorization	
Preferred QOP	<p>Specifies the quality of protection mode to use when the server offers multiple choices. Available choices are:</p> <ul style="list-style-type: none"> • auth(entication) (default) • auth(entication)-int(egrity)
Treat parser warnings as errors	<p>If selected, the SIP parser warnings are considered errors. The default is Disabled.</p>
Messages Queue Settings	
Remove previously received messages	<p>If set to Yes, clears the message queue of previously received messages. The default is No.</p>
Limit queue size to maximum	<p>If selected, this option allows you to indicate the maximum number of messages per channel that the queue can sustain.</p>
Log	<p>For each voip peer activity, if any of the logs are enabled, a file is created and is named 'voip_peer_\$activityName.log.'</p>
SIP sent Messages	<p>Check this option to log all SIP sent Messages.</p>

SIP received Messages	Check this option to log all SIP received Messages.
SIP matched Messages	Check this option to log all SIP matched Messages.
Scenario Flow Path	Check this option to log all Scenario functions entry and exit.
Timeline Events	Check this option to log timeline events for each user.
Call States	Check this option to log the SIP Call States.
RTP/SDP informations	Check this option to log all RTP/SDP events.
Call/Registration times	Check this option to log all SIP times.
Errors	Check this option to log all SIP errors.
Log for channels	Check this option to specify the channels that are logged.
Include timestamps	Check this option to include timestamps for the logs.

STUN Settings

Because protocols such as SIP/RTP/RTCP use UDP packets for the transfer of sound, video, and real-time signaling traffic over the Internet, SIP endpoints operating from behind NAT devices have to use a STUN server in order to allow inbound media traffic. A STUN server enables endpoints behind a NAT device to first discover the presence and the type of a NAT, and then learn the address bindings allocated by the NAT.

The value of the **Send STUN packets at every** field defines the keep-alive time for the sent STUN packets.

T.38 Settings

The following table lists the global T.38 test library settings:

Name	Description
T.38 Log Settings - This group specifies settings related to the collection of logs for T.38 traffic.	
Enabled logs	Specifies the logs to be collected when executing a T.38 Fax Session script function: <ul style="list-style-type: none"> • T30 signals • T38 signals • Received image: If selected, received images are collected and stored on the Ixia ports in the /var/log folder following a naming pattern that includes the activity name, the user name, and the current loop, such as for example in /var/log/VoIPSipPeer1_recv_img_user003_loop001.tif
Log type	Specifies if logs are collected for a number of loops (Loops option), or for a specified duration of time (Duration , expressed in minutes).

PSTN Settings

The following table lists the global T1/E1 test library settings:

Parameter	Description
Delay before execution	Specifies a global function execution delay value as either: <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms)
No answer timeout	Specifies the period of time for which the Make Call script function waits for an answer, after which the function exits on the Timeout output.
Wait other party to disconnect	If selected, the End Call script function does not initiate call termination itself, but waits for the other party to disconnect.
Wait for call timeout	Specifies a period of time a Receive Call function waits for an incoming call, after which it exits on the Timeout output.
Completion timeout	Specifies timeout value for the initiation of a call to the actual call established status.

PSTN Library - ISDN

The following table lists the global T1/E1 test library settings:

Parameter	Description
Layer 1 bearer channel	The User info layer 1 parameter specified by ITU-T Recommendation Q.931, which can take the following values: <ul style="list-style-type: none"> • ISDN_UIL1_G711ULAW • ISDN_UIL1_G711ALAW • OPERATOR_SPECIFIC
Bearer channel capability	The Information Transfer Capability parameter specified by ITU-T Recommendation Q.931, which can take the following values: <ul style="list-style-type: none"> • BEAR_CAP_SPEECH • BEAR_CAP_UNREST_DIG • BEAR_CAP_DOT1K_AUDIO

Destination number type	<p>The called party number information element specified by ITU-T Recommendation Q.931, which can take the following values:</p> <ul style="list-style-type: none"> • Unknown • International • National • Network specific • Subscriber • Abbreviated
Destination number plan	<p>The called party Numbering plan identification parameter specified by ITU-T Recommendation Q.931, which can take the following values:</p> <ul style="list-style-type: none"> • Unknown • ISDN (default) • Telephony • National • Private
Originating number type	<p>The caller party number information element specified by ITU-T Recommendation Q.931, which can take the following values:</p> <ul style="list-style-type: none"> • Unknown • International • National • Network specific • Subscriber • Abbreviated
Originating number plan	<p>The caller party numbering plan identification parameter specified by ITU-T Recommendation Q.931, which can take the following values:</p> <ul style="list-style-type: none"> • Unknown • ISDN (default) • Telephony • National • Private
Reject call reason	<p>Specifies a global call reject reason for the Receive Call script function. When this function rejects an incoming call, the configured reason is provided to the calling party.</p>
Initiate RESTART procedure while shutting down spans	<p>If selected, a RESTART procedure is initiated when the ISDN protocol is stopped. This procedure implies sending RESTART messages to the remote end (a single RESTART for the entire trunk or one RESTART for each B channel, depending on the variant) and waiting for RESTART ACKNOWLEDGE messages.</p>

PSTN Library - CAS

The following table lists the global T1/E1 test library settings:

Parameter	Description
Dial digits	
Digit duration	Specifies the digit duration.
Inter digit delay	Specifies the inter-digit delay value.
Signal power	Specifies the CAS signal power value.
ANI/DNIS	
Incoming call : Digits format	The digits format for incoming calls, which can be either of the following: <ul style="list-style-type: none"> • DNIS • DNIS/ANI • ANI/DNIS
Incoming call : Calling party # of digits (ANI)	The expected number of digits that identify the calling party. <div style="border: 1px solid gray; padding: 5px; margin-top: 5px;"> <p>! Important! If the calling party number is configured to a higher value at activity level, that value is used.</p> </div>
Outgoing call : Digits format	The digits format for outgoing calls, which can be either of the following: <ul style="list-style-type: none"> • DNIS • DNIS/ANI • ANI/DNIS
Timeouts	
Wait for wink timeout	The timeout value for receiving a wink.
Time between offhook and dial	The maximum period of time between the offhook signal and the first dialed digit.

PSTN Library - Voice

The following table lists the global T1/E1 test library settings:

Parameter	Description
Playback	


Play x time(s)	The number of times the wave is played.
Repeat Continuous for	The period of time the wave is played for (in seconds, minutes, or hours).
Use Talk Time (all objectives except Channels)	In the case of a CPS objective, choosing this option plays back the wave for the duration of the talk time parameter.
Coding	
Data format	The voice encoding, any of the following: ALaw, MuLaw, PCM.
Sampling rate	The voice sampling rate, 8000 or 11025 Hz.
Bits/sample	The number of bits per voice sample, 8 or 16 bit.
DTMF/MF/Tones Generation	
Tone Duration	The time duration (in ms) of a single tone. The range of values is 50 to 10000 ms. The default value is 200 ms.
Inter Tone Interval	The maximum amount of time (in ms) between two consecutive generated DTMFs. The range of values is 50 to 10000 ms. The default value is 200 ms.
Tone Amplitude	The attenuation (in dB) of the DTMF tone. The minimum attenuation is -3 dB (no attenuation) and the maximum is -54 dB. The default value is -3 dB.
DTMF/MF/Tones Detection	
First tone timeout	The time (in ms) allowed for receiving the first tone. After this period elapses, the function exits on the Timeout output. The range of values is 50 to 10000 ms. The default value is 3000 ms.
Max delay between tones	The maximum delay (in ms) between subsequent tones. elapses, the function exits on the Timeout output. The range of values is 50 to 10000 ms. The default value is 2000 ms.

Function Output Results


The **Function Outputs Results** window enables you to define the global result resolution (the output name—output result association) used by Voice functions.

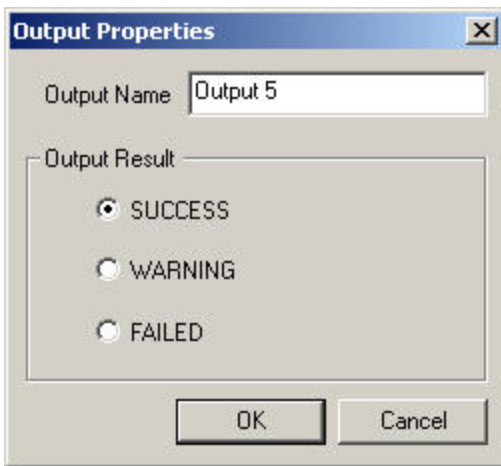
All Voice functions placed in the Scenario Editor have a special Outputs tab that enables you to set the result resolution for each output. When a function output is set to the **GLOBAL** value, it takes the value corresponding to the specified output in the **Function Outputs Results** window.

To configure the global settings for each output, do the following:

1. Click **Add** . The **Output Properties** dialog box appears.
2. Type the required **Output Name** in the appropriate field.
3. Choose the **Output Result**. You have the following options:
 - SUCCESS
 - WARNING
 - FAILED
4. Click **OK** to add the new output.


To modify an existing output, do the following:


1. Select the required output in the list and click Edit , or double click the required output in the list. The **Output Properties** dialog box appears, as shown in the following image:



2. Make the required changes to the output parameters. Note that you are not allowed to modify the name of a predefined output.
3. Click **OK** to save the settings.

To remove an existing output, do the following:

1. Select the output to remove.
2. Click **Delete** . The output is deleted from the list.

 **Note:** The predefined outputs cannot be removed.

To save your Test Library Settings and Outputs configuration to a file, do the following:

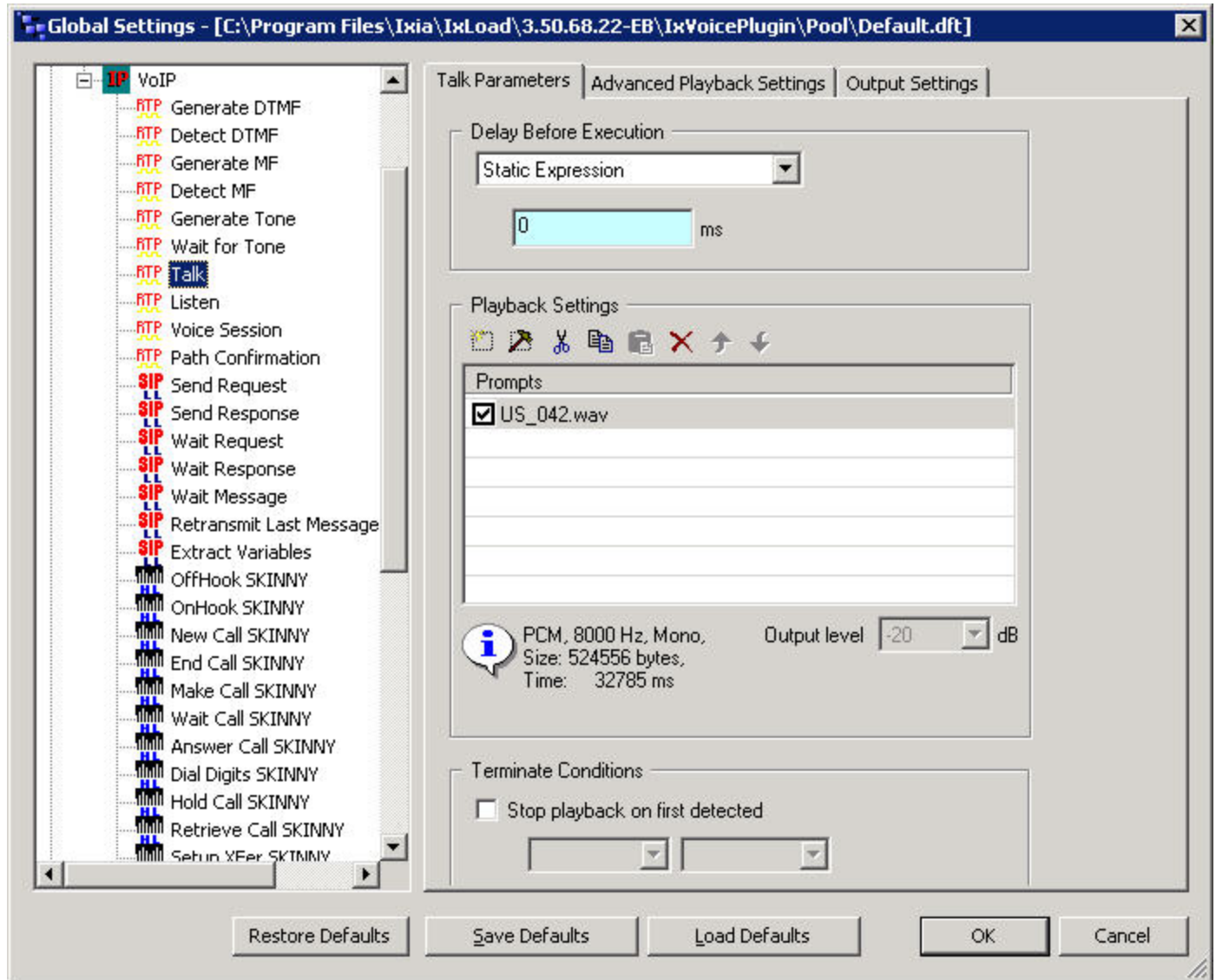
1. Click **Save Library Settings**. The **Save As** dialog box appears.
2. Choose the location and type in the file name, and then click **Save**. The library settings are saved to a file with the .lsf extension (Library Settings File).

To load an existing configuration for the test libraries and outputs, do the following:

1. Click **Load Library Settings**. The **Open** dialog box appears.
2. Browse for the file where the desired execution settings are stored (an .lsf file) and open it. The settings are applied to your configuration.

Configuring Scenario Editor Default Settings

Click **Scenario Editor Defaults** in the left pane of the Global Settings window to define the default parameters for each SIP, Skinny, H.323, MGCP, H.248/MEGACO, Digital T1/E1, and RTP test library function.



Note: When placing a script function in the Scenario Editor, its parameters are those set in the Scenario Editor Default window. These default parameters can be changed at any time by double-clicking the function body and editing as required.

Note: The parameters available for each script function are described in detail in [Voice Functions Reference](#).

The function parameter configuration can be saved to a file, which enables you to reapply the configuration at a later time.

To save the default parameters values to a file:

1. Click **Save Defaults**. The **Save As** dialog box appears.
2. Choose the location and type in the file name, and then click **Save**. Note that the file has the .dft extension (Defaults Settings File).

To load the default values from a file:

1. Click **Load Defaults**. The **Open** dialog box appears.
2. Browse for the file where the required execution settings are stored (a .dft file) and open it. The settings are applied to your configuration.

To restore the default values for a function, select the function in the left tree pane and click **Restore Defaults**.

This page intentionally left blank.

CHAPTER 3 Voice Functions Reference

This section describes the script functions in the Voice Test Libraries.

In this chapter:

- VoIP SIP Functions Library 32**
- VoIP Skinny Functions Library 65**
- VoIP Media Functions Library 95**
- VoIP Flow Functions Library 119**
- VoIP H323 RAS Library 124**
- VoIP H248 Functions Library 127**
- VoIP HTTP Functions Library 139**
- VoIP MGCP Functions Library 145**
- Digital T1/E1 Functions Library 160**

VoIP SIP Functions Library

The VoIP SIP Test Library contains the following script functions:

- [Send Request](#)
- [Send Response](#)
- [Wait Request](#)
- [Wait Response](#)
- [Wait Message](#)
- [Retransmit Last Message](#)
- [Extract Variables](#)
- [T.38 Fax Session](#)
- [MSRP Session](#)
- [MSRP Control](#)


Send Request


Sends a SIP request.

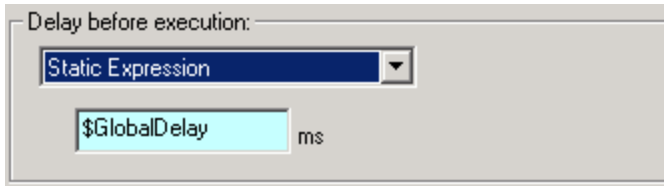
Send RT.38request Properties: Parameters

The following describes the Send Request function parameters:

Name	Description
SIP Message	The SIP message to transmit. It can be edited manually in the message preview window, imported from a text file, or generated using the Template GUI.
Load from file	Loads the SIP message to transmit from an existing text file.
Create From Template	Generates a correct SIP message based on existing message templates, as described in Creating a SIP Message from Template in Appendix A.
Edit Options	
Case Sensitive	Enables/disables case-sensitivity for the SIP message content. By default, case-sensitivity is disabled.
Change Case	If enabled, the SIP method names are written in uppercase letters (and in blue color) in the SIP message. This option works only if the Case Sensitive option is disabled. By default, it is enabled. Available SIP methods are INVITE, ACK, BYE, CANCEL, OPTIONS, REGISTER, MESSAGE, NOTIFY, SUBSCRIBE, REFER, PRACK, INFO, UPDATE.



Delay Before Execution	<p>Delays the function execution by a value specified as:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 2000, in milliseconds (ms)
Message Body	<p>Defines the SDP information sent in the SIP message body:</p> <p>Send Audio SDP – Enables/disables sending audio SDP information. When this option is selected, the SDP can be transmitted as:</p> <ul style="list-style-type: none"> • Offer – (default) Sends the SDP information as defined in the test/activity configuration by the VoIP activity codecs list in the Codec Settings page. • Negotiated – Sends the SDP information as negotiated between the SIP endpoints. • Hold Session – Sends the SDP information such as to establish a hold session. <p>Send Custom Message Body – Enables/disables sending custom messages.</p> <p>Clicking Edit Custom opens the Custom Message Body window, which enables you to define the message body by freely editing it, or based on a predefined template. You can also add multipart message bodies, as described in Support for Multipart SIP Messages in Appendix E.</p> <p>The following options are available in the Custom Message Body window:</p> <ul style="list-style-type: none"> • Evaluate Expression between character(s) – the expression between the user-defined characters is evaluated. • Extract SDP Information – the custom body is parsed by the SDP parser to extract SDP Offer/Answer media streaming information. Selecting this option implies updating, adding, or deleting media streams for the current endpoint; otherwise, the content body is ignored from the SDP Offer/Answer exchange point of view. • Send unmodified SDP in case of negotiation – Sends the complete capabilities information in case of negotiation. • Show CR/LF – Shows the CR/LF character in the SIP message body. • Create from template – Creates a custom message body based on one of the following predefined templates: Simple Session, Simple Secure Session, Multiple Sessions, G723 at 5.3 Bitrate, G729 Annex A, AMR Octet Aligned. <p>Clicking Variables enables you to access scenario variables for use in the SIP custom message body.</p> <hr/> <p> Note: For the media to be correctly set up, the \$VOIP_MediaIP and \$VOIP_MediaBasePort scenario variables are used in the custom message body.</p>

 **Note:** Fields highlighted blue (shown in the image below) in the script function configuration tabs indicate that expressions using scenario variables and numerical values are accepted as input in these fields.



Send Request Properties: Behavior

This page allows you to specify the destination for the SIP message, as well as the parameter values, the transport protocol to use, and the dialog layer (new/existing dialog). The following table describes the Send Request Behavior parameters:

Parameter	Description
Auto-Variables	Allows you to set the values for the SIP auto variables.
Custom Behavior	<p>If selected, allows you to override the destination for the SIP message by specifying a destination address and port.</p> <hr/> <p> Note: If this option is enabled, all the other settings in the SIP message are ignored.</p> <hr/>
Transport Layer	<p>Sets the function-level transport protocol to use. You have the following options:</p> <ul style="list-style-type: none"> • TCP • UDP • Last sent transport type – It uses the same protocol used by the last sent message. If it cannot be evaluated, the default is UDP. • Last received transport type – It uses the same protocol used by the last received message. If it cannot be evaluated, the default is UDP.
Dialog Layer	<p>Specifies if the script function generates a new SIP dialog or uses the existing dialog. In SIP terms, a dialog is a SIP communication established between two UAs that persists for some time. It is established by SIP messages such as a 2xx response to an INVITE request, and is identified by a call identifier, a local tag, and a remote tag.</p>
Remove previously received messages	<p>If selected, the message queue is emptied at function execution time. When the Global option is selected, the value from the Global Settings > Library Settings and Outputs > VoIP > SIP page is used.</p>
Extended variables support	<p>If selected, enables support for test scenario variables, both read-only and custom, used in SIP message headers. With variable support enabled, at function execution time scenario variables are evaluated and replaced with the actual values.</p> <hr/> <p> Note: This option does not affect auto variables.</p> <hr/>

Send Request Properties: Flow Manager

By overriding activity-level settings such as destination or transfer addresses, the settings in this page control the way the SIP message is transmitted. Available options are described in the following table:

Parameter	Description
Send this message based on	Defines the transmission mode, using either configured activity level settings or overriding settings specified in this page.
Default Settings	The message is transmitted using the predefined activity settings.
Redirect Information	<p>The message is transmitted using redirect information included in the message. Current implementation limits the number of redirect information to three addresses, referenced as First, Second, or Third address.</p> <p>If there is no redirect info available, the function returns an error message. If an IP address is specified exceeding the number of redirect info items (for example, you choose Third Address and only two redirects are defined), the First Address is automatically selected.</p>
Raw Information	<p>The message is transmitted as is, with the auto variables not being evaluated. Available options are:</p> <ul style="list-style-type: none"> • Using Test/Activity settings – The message uses the activity settings. • Using DNS or IP address – The message uses the specified IP address or DNS name and port (mandatory). If the DNS name is used, the application checks it using the current network settings on the local machine. A warning message appears if the address cannot be resolved on the network. <p>Evaluate the expressions in the proper fields – If enabled, expression(s) contained in any of the above fields are evaluated.</p>

<p>Semantic Information</p>	<p>You can choose between:</p> <ul style="list-style-type: none"> • Using Test/activity settings – It uses the settings in the activity configuration. • Using DNS or IP address – It uses the specified IP address or DNS name and port (mandatory). If the DNS name is used, the application checks it using the current network settings on the local machine. A warning message appears in case the address cannot be resolved on the network. When choosing this option, a proxy or registrar role can be selected for the IP address. <p>You can override the activity level Destination Phone, Destination IPAddress, Destination Port, and TransferAddress settings by using variables or expressions in the fields below.</p> <p>When the activity is contained in a VoIPSIP cloud, you can also override the activity level Source Address and Source Port, default values being the address and port of the first SIP Proxy Server that is part of the cloud.</p> <p>Evaluate the expressions in the proper fields – If enabled, expression(s) contained in any of the above fields are evaluated.</p>
<p>Retransmission Settings</p>	<p>Specifies retransmission settings that override the activity level settings defined in the VoIP Plug-in SIP Settings tab.</p>
<p>Follow settings specified at activity level</p>	<p>When selected, the activity-level retransmission settings are used. You can, however, specify different T1 and T2 timer values by entering values in the fields below.</p>
<p>Disable retransmission</p>	<p>When selected, the message is not retransmitted, regardless of the activity level retransmission settings.</p>

Send Request Properties: Extract Variables


This page allows you to select the SIP variables you want to be extracted from the SIP message.

The SIP test library also comprises a standalone script function that performs variable extraction, as described in [Extract Variables](#).

Send Request Properties: Authentication

The available authentication settings are described in the following table:

Parameter	Description
-----------	-------------

Override UAC Side Authentication	<p>If enabled, the authentication credentials are overwritten with information specified in the fields described below:</p> <ul style="list-style-type: none"> • Realm • User Name • Password <p>These fields accept string values (written between quotation marks) and variables (prefixed by the '\$' character).</p>
Digest	<p>Preferred QOP: Specifies the quality of protection level to use when the server offers multiple choices. Available choices are:</p> <ul style="list-style-type: none"> • auth(entication) • auth(entication)-int(egrity) • global (default) <p>When the Global option is selected, the value from the Global Settings > Library Settings and Outputs > VoIP > SIP page is used.</p> <hr/> <p> Note: The auth-int setting may lead to possible problems when traversing Network Address Translators (NATs), Back-to-Back UAs (B2BUAs) and Application Level Gateways (ALGs), any of which may modify the body to permit the SIP request to traverse some form of network boundary. In this case, the NAT/B2BUA/ALG must also act as an endpoint and police and possibly modify the authentication header.</p>

Send Request Properties: Outputs

The following table describes the outputs available for the Send Request function:

Parameter	Description
OK	The function completed successfully. The default resolution for this output is SUCCESS.
Transport Failure	The default resolution for this output is WARNING.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

Send Response

Sends a SIP response.

Send Response Parameters

The following table describes the Send Response function parameters:

Parameter	Description
-----------	-------------


SIP Message	The text SIP message to transmit. It can be edited manually in the message preview window, imported from a text file, or generated using the Create from Template GUI.
Load from File	Loads from a text file the SIP message to transmit.
Create From Template	Generates the SIP message based on existing message templates. For more details on this window enabling you to create a correct SIP message, see Creating a SIP Message from Template in Appendix A.
Edit Options	
Case Sensitive	Enables or disables case sensitivity of the SIP message.
Change Case	If enabled, the SIP method names are written in uppercase letters (and in blue color) in the SIP message. This option works only if the Case Sensitive option is disabled. By default, it is enabled. Available SIP methods are INVITE, ACK, BYE, CANCEL, OPTIONS, REGISTER, MESSAGE, NOTIFY, SUBSCRIBE, REFER, PRACK, INFO, UPDATE.
Delay Before Execution	Delays the function execution by a value specified as: <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 2000, in milliseconds (ms)

Message Body	<p>Defines the SDP information sent in the SIP message body:</p> <p>Send Audio SDP – Enables/disables sending audio SDP information. When this option is selected, the SDP can be transmitted as:</p> <ul style="list-style-type: none"> • Offer – (default) Sends the SDP as defined in the test/activity configuration by the VoIP activity codecs list in the Codec Settings page. • Negotiated – Sends the SDP information negotiated between two SIP endpoints. • Hold Session – Sends SDP such as to establish a hold session. <p>Send Custom Message Body – Enables/disables sending custom messages. You can also add multipart message bodies, as described in Support for Multipart SIP Messages in Appendix E.</p> <p>Click Edit Custom to open the Custom Message Body window, which enables you to define the message body by freely editing it, or based on a predefined template.</p> <p>The following options are available in the Custom Message Body window:</p> <ul style="list-style-type: none"> • Evaluate expression between character(s) – the expression between the user-defined characters is being evaluated. • Extract SDP Information – the custom body is parsed by the SDP parser to extract SDP Offer/Answer media streaming information. Selecting this option implies updating, adding, or deleting media streams for the current endpoint; otherwise, the content body is ignored from the SDP Offer/Answer exchange point of view. • Send unmodified SDP in case of negotiation: Sends the complete capabilities information in case of negotiation. • Show CR/LF – Shows the CR/LF character in the SIP message body. • Create from template – Creates a custom message body based on one of the following predefined templates: Simple Session, Simple Secure Session, Multiple Sessions, G723 at 5.3 Bitrate, G729 Annex A, AMR Octet Aligned. <p>Click Variables to access scenario variables for use in the SIP custom message body.</p>
--------------	--

Send Response Properties: Behavior

This section allows you to specify the destination of the SIP message, as well as the parameter values and the transport protocol to use. The following table describes the Send Response Behavior parameters:

Parameter	Description
Auto-Variables	Allows you to set the values for the SIP auto variables.
Custom Behavior	Allows you to specify the destination of the SIP message using custom value for destination address and destination port.

Transport Layer	<p>Sets the function-level transport protocol to use:</p> <ul style="list-style-type: none"> • TCP • UDP • Last sent transport type – It uses the same protocol used by the last sent message. If it cannot be evaluated, the default is UDP. • Last received transport type - It uses the same protocol used by the last received message. If it cannot be evaluated, the default is UDP.
Dialog Layer	<p>Specifies if the script function uses the existing dialog or creates a new one. A dialog is a peer-to-peer SIP relationship between two UAs that persists for some time. A dialog is established by SIP messages, such as a 2xx response to an INVITE request, and is identified by a call identifier, a local tag, and a remote tag.</p>
Remove previously received messages	<p>If selected, the message queue is emptied at function execution time. When the Global option is selected, the value from the Global Settings > Library Settings and Outputs > SIP page is used.</p>
Extended variables support	<p>If selected, enables support for test scenario variables, both read-only and custom, used in SIP message headers. With variable support enabled, at function execution time scenario variables are evaluated and replaced with the actual values.</p> <hr/> <p> Note: This option does not affect auto variables.</p>

Send Response Properties: Flow Manager


The settings in this page control the way the SIP message is transmitted. The available options are described in the following table:

Parameter	Description
Send this message based on	<p>Defines the message transmission mode, using either configured activity-level settings or overriding settings specified in this page.</p>
Default Settings	<p>The message is transmitted with the default activity settings (default).</p>
Raw Information	<p>The message is transmitted as is, with the auto variables not being evaluated. The available options are as follows:</p> <ul style="list-style-type: none"> • Using Test/Activity settings – The message uses the activity settings. • Using DNS or IP address – The message uses the specified IP address or DNS name and port (mandatory). If the DNS name is used, the application checks it using the current network settings on the local machine. A warning message appears if the address cannot be resolved on the network. <p>Evaluate the expressions in the proper fields – If enabled, expression(s) contained in any of the above fields are evaluated.</p>

Retransmission Settings	Specifies retransmission settings that override the activity level settings defined in the VoIP Plug-in SIP Settings tab.
Follow settings specified at activity level	When selected, the activity-level retransmission settings are used. You can, however, specify different T1 and T2 timer values by entering values in the fields below.
Disable retransmission	When selected, the message is not retransmitted, regardless of the activity-level retransmission settings.

Send Response Properties: Extract Variables

This section allows you to select the SIP variables you want to be extracted from the SIP message.

 **Note:** The SIP test library also comprises a standalone script function that performs variable extraction, as described in [Extract Variables](#).

Send Response Properties: Outputs


For more information, see [Send Request Properties: Outputs](#).

Wait Request

Waits for a SIP request.

Wait Request Parameters


The following table describes the Wait Request function parameters:

Name	Description
SIP Message	The SIP request for which to wait, which can be generated using the Create from Template GUI.
Create From Template	Generates the SIP message based on existing message templates. For details on this window enabling you to create a correct SIP message, see Creating a SIP Message from Template in Appendix A.
Ignore SDP	If selected, the message body is ignored from the SDP offer/answer exchange point of view.
Extended Variables Support	If selected, enables support for test scenario variables, both read-only and custom, used in SIP message headers. With variable support enabled, at function execution time scenario variables are evaluated and replaced with the actual values. <hr/>  Note: This option does not affect auto variables.

Delay Before Execution	<p>Delays the function execution by a duration that can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, Sleep 2000 in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms)
Timeout	<p>The time (in ms) the function waits for the received message to match the template. If this time period terminates without having a match, the function enables the Timeout output. It can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Delay Between Digits, in milliseconds (ms) • PHONE_WAIT_TIME, in milliseconds (ms) • MGCP Timeout, in milliseconds (ms)
Remove previously received messages	<p>If selected, the message queue is emptied at function execution time. When the Global option is selected, the value from the Global Settings > Library Settings and Outputs > VoIP > SIP page is used.</p>


Wait Request Properties: Extract Variables

This section allows you to select the SIP variables that you want to be extracted from the SIP message.

 **Note:** The SIP test library also comprises a standalone script function that performs variable extraction, as described in [Extract Variables](#).

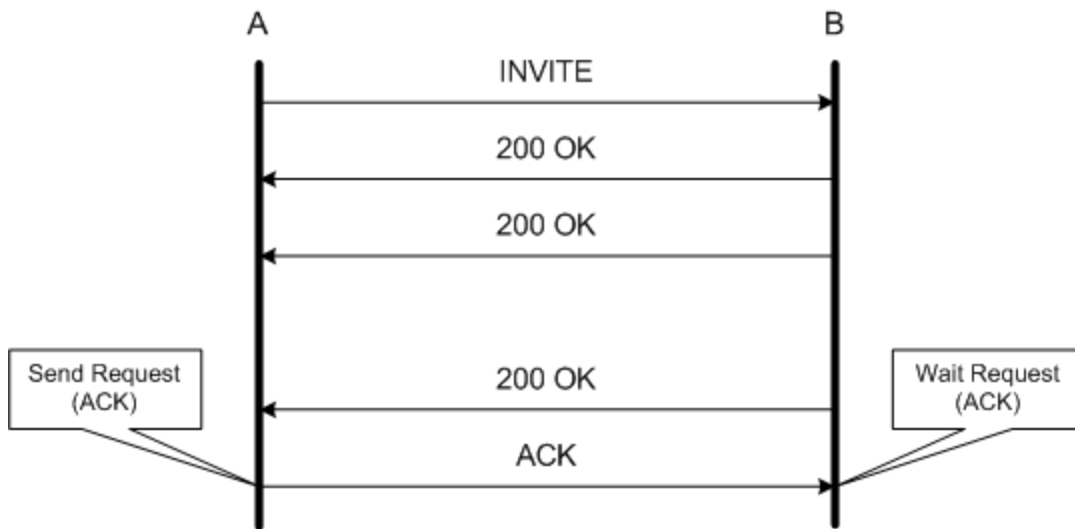
Wait Request Properties: Retransmission

The following table describes the retransmission settings available for the Wait Request function:

Name	Description
Retransmission Settings	<p>When a request message comprised in a SIP transaction is matched, these settings define the conditions for ending or keeping transaction-level retransmissions of messages.</p> <p> Note: For each script function that has retransmissions enabled and carries a SIP message subject to retransmissions, a retransmission rule is created. Each scenario channel maintains a list of known retransmission rules, one for every such script function that started retransmissions.</p>

Clean matched retransmission rule	If selected, the matched retransmission rule is stopped.
Keep selected retransmission rules active (and clean the other)	If selected, from all known retransmission rules that are shown in the Started in Function list, only the selected rule(s) remain(s) active.

For example, considering the transaction messages flow shown in the image below:



whereby an INVITE message is sent by endpoint A, and endpoint B retransmits a response 200 OK message for a number of times at increasing intervals.

Considering the retransmission settings for the Wait Request (ACK) script function executed by endpoint B, the following behaviors can be implemented depending on the retransmission settings:

- If the **Clean matched retransmission rule** option is selected, the 200 OK message retransmission is stopped after receiving the ACK request.
- If the **Keep selected retransmission rule active** option is selected and the **Send 200 OK** rule is checked, the 200 OK message is retransmitted after receiving the ACK request, until the retransmission timeout expires.

Wait Request Properties: Outputs

The following table describes the outputs available for the Wait Request function:

Output Name	Description
OK	The function completed successfully. The default resolution for this output is SUCCESS.


Timeout	This output is enabled if the request is not received within the specified timeout. The default resolution for this output is WARNING.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

Wait Response

Waits for a SIP response.

Wait Response Parameters

The following table describes the Wait Response function parameters:


Name	Description
SIP Message	The SIP response for which to wait. It can be generated using the Create from Template GUI.
Create From Template	Generate the SIP message based on existing message templates. For more details on this dialog that helps you create a correct SIP message, see Creating a SIP Message from Template in Appendix A.
Ignore SDP	If selected, the message body is ignored from the SDP offer/answer exchange point of view.
Extended Variables Support	If selected, enables support for test scenario variables, both read-only and custom, used in SIP message headers. With variable support enabled, at function execution time scenario variables are evaluated and replaced with the actual values.  Note: This option does not affect auto variables.
Delay Before Execution	Delays the function execution by a duration that can be specified as follows: <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, Sleep 2000 in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms)
Timeout	The time (in ms) the function waits for the received message to match the template. If this time period terminates without having a match, the function enables the Timeout output. It can be specified as follows: <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Delay Between Digits, in milliseconds (ms) • PHONE_WAIT_TIME, in milliseconds (ms) • MGCP Timeout, in milliseconds (ms)

Wait Response Properties: Extract Variables

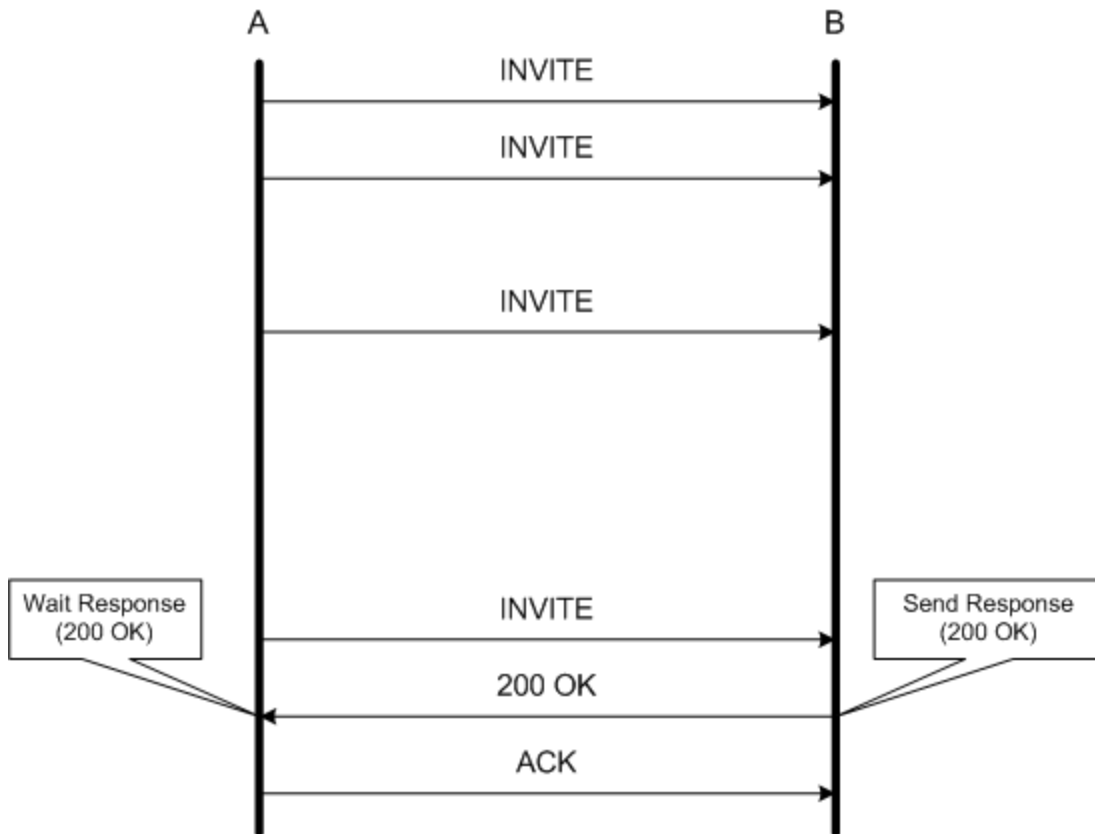
This page allows you to select the SIP variables you want to be extracted from the SIP message.

Wait Response Properties: Retransmission

The following table describes the retransmission settings available for the Wait Response function:

Parameter	Description
Retransmission Settings	<p>When a request message comprised in a SIP transaction is matched, these settings define the conditions for ending or keeping transaction-level retransmissions of messages.</p> <hr/> <p> Note: For each script function that has retransmissions enabled and carries a SIP message subject to retransmissions, a retransmission rule is created. Each scenario channel maintains a list of known retransmission rules, one for every such script function that started retransmissions.</p> <hr/>
Clean matched retransmission rule	If selected, the matched retransmission rule is stopped.
Keep selected retransmission rules active (and clean the other)	If selected, from all known retransmission rules shown in the Started in Function list, only the selected rule(s) remain(s) active.

For example, considering the transaction messages flow shown in the image below:



whereby an INVITE message is sent repeatedly for a number of times (retransmitted) by endpoint A, and endpoint B transmits a response 200 OK message.

Considering the retransmission settings for the Wait Response (200 OK) script function executed by endpoint A, the following behaviors can be implemented:


- If the **Clean matched retransmission rule** option is selected, the INVITE message retransmission is stopped after receiving the 200 OK response.
- If the **Keep selected retransmission rule active** option is selected and the **Send INVITE** rule is checked, the INVITE message is further retransmitted after receiving the 200 OK response, until the retransmission timeout expires.

Wait Response Properties: Outputs

For further information, see [Wait Request Properties - Outputs](#).

Wait Message

Waits for a SIP request or response message and matches it against one of the message templates defined in the script function.

 **Note:** The Wait Message function adds flexibility to scenario configuration, by enabling you to receive and match multiple messages using a single script function, instead of multiple Wait Request/Wait Response ones.






Wait Message Properties: Templates




The received message is compared with each template from a list of up to 10 templates, starting with the template at the top of the list.

Comparison of the received message with a message template is based on:

- The template type (request, response)
- Request options: Request Line Method (INVITE, ACK, OPTIONS,...)
- Response options: Status Code Value (200 (OK), 182, ...)
- For each template, a logical scheme may be defined at header level (Call-Id, Contact, CSeq, From, To). If a header is not checked, the Wait Message function does not take it into consideration.

In case of a match, the function enables the output corresponding to the matched template, or a Timeout output if no match was found. The Wait Message available options are listed in the following table:

Option	Description
Template Name	User-defined template name in the list. By default, names with the msg_name#01, msg_name#02,format are assigned.
Template Type	The type of message for which to wait. The available options are as follows: - (request) - (response) For more details about these message types, see The Template Window .
 New Template	Creates a new template, configured using the Template window, as described in The Template Window .
 Delete Template	Deletes the selected templates (supports multiple selections).
 Move Up	Moves the selected templates one position up (supports multiple selections).
 Move Down	Moves the selected templates one position down (supports multiple selections).  Note: Moving up/down the headers establishes the order in which they are matched by the Wait functions with the defined templates.

 Preview	<p>Opens the preview window that displays the SIP message headers for the selected message.</p> <hr/> <p> Note: To modify an existing message template, doubleclick the template to open the Template editor window.</p>
Wait for any message	<p>If enabled, the Wait Message function exposes an additional output, Other, that is enabled if the received message does not match any of the defined SIP message templates.</p> <p>When this option is disabled, the function enables the Timeout output if the received message does not match any of the defined SIP message templates (default).</p> <hr/> <p> Note: This option can be selected only if there is at least one template defined in the Templates list.</p>
Message headers	The general headers included with the selected template. For more details, see Message Header Parameters .
Ignore SDP	If selected, the message body is ignored from the SDP offer/answer exchange point of view.
Observations:	User-defined field with a comment referring to the selected template, which does not influence the function behavior.
Restore Defaults	Sets the function parameters to their default settings, as specified for the Wait Message function in the Global Settings > Scenario Editor Defaults > VoIP > Wait Message page.
OK	Applies the settings and closes the Properties window.
Cancel	Discards the changes and closes the Properties window.
Apply	Applies the settings.

The Template Window

The following table describes the **Template** window parameters for the Wait Message function:

Option	Description/Available Values
Template Name	User-defined template name. By default, names with msg_name#01, msg_name#02,...formats are assigned.
Message Type:	Type of SIP message for which to wait. The available types are as follows: <ul style="list-style-type: none"> - Request (default) - Response

Request options	<p>Request Line Method</p> <p>The message template for which to wait. The available options are as follows:</p> <ul style="list-style-type: none"> • INVITE • ACK • OPTIONS • BYE • CANCEL • REGISTER • REFER • NOTIFY • SUBSCRIBE • MESSAGE • REFER • PRACK • INFO • UPDATE
Response options	<p>Status Code – the status code for which to wait.</p>

Message Header Parameters

After a template has been added to the list, message header parameters must be edited according to the following conventions for the Value column content:

- |ANY| – the corresponding parameter must be present and may take any value. This is the matching condition for the parameter in the received message.
- |NOT_ALLOWED| – the corresponding parameter must not be present. This is the matching condition for the parameter in the received message.
- n/a – the corresponding parameter has no available options/values.
- User-defined field – this is the comment for the parameters that support user defined values. The default value is also specified.

The other lines in the Value column are SIP options. For more details, see RFC 3261 (SIP).

The following table describes the Message Headers parameters:

Header/Parameters	Available Options/Values
Call-ID	Call-ID i ANY NOT_ALLOWED

Call-Id	Contact m ANY NOT_ALLOWED
STAR	n/a
contact-param	n/a
name-addr	ANY_NAME_ADDR NOT_ALLOWED_NAME_ADDR
addr-spec	ANY_ADDR_SPEC NOT_ALLOWED_ADDR_SPEC
Param	Q expires ANY_PARAM NOT_ALLOWED_PARAM
Value	User-defined field. The default is an empty string.
Cseq	Cseq ANY NOT_ALLOWED
Numeric Value	User-defined field. The default is 0.
Method	INVITE ACK OPTIONS BYE CANCEL REGISTER REFER NOTIFY SUBSCRIBE MESSAGE PRACK INFO UPDATE The default is an empty string.


From	From f ANY NOT_ALLOWED
name-addr	ANY_NAME_ADDR NOT_ALLOWED_NAME_ADDR
addr-spec	ANY_ADDR_SPEC NOT_ALLOWED_ADDR_SPEC
Param	Tag ANY_PARAM NOT_ALLOWED_PARAM
Value	User-defined field. The default is an empty string.
To	To t ANY NOT_ALLOWED
name-addr	ANY_NAME_ADDR NOT_ALLOWED_NAME_ADDR
addr-spec	ANY_ADDR_SPEC NOT_ALLOWED_ADDR_SPEC
Param	Tag ANY_PARAM NOT_ALLOWED_PARAM
Value	User-defined field. The default is an empty string.

Check the header(s) that you want to be taken into consideration. Uncheck the header(s) that you do not want to be taken into consideration.

Wait Message Properties: Params

The following table describes the Wait Message Properties parameters:

Name	Description
------	-------------


Extend Variables Support	<p>If selected, enables support for scenario variables—both read-only and custom—used in SIP message headers. With variable support enabled, at function execution time, scenario variables are evaluated and replaced with the actual values.</p> <hr/> <p> Note: This option does not affect auto variables.</p>
Delay Before Execution	<p>Delays the function execution by a duration that can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – Fixed Delay Before Execution duration. It can be used for synchronization reasons. It can be a value in milliseconds (ms), or a formula. • Random Between Expressions – Random Delay Before Execution duration value in the specified interval. It can be used to simulate real-life conditions. • A user-defined delay (VoIP constant), chosen from those available in the Resource Pool VoIP Constants. <p>The default is Static Expression, 0 ms.</p>
Timeout	<p>The time (in ms) the function waits for a specific incoming message defined by templates. If this time period terminates without matching any of the defined message templates, the function exits on the Timeout output. It can be specified as follows:</p> <ul style="list-style-type: none"> • Static expression/value, in milliseconds (ms) • Random between two values, in milliseconds (ms) • A user-defined timeout (VoIP constant), chosen from those available in the Resource Pool > VoIP Constants. <p>The default is Static Expression, 20000 ms.</p>
Remove previously received messages	<p>If selected, the message queue is emptied at function execution time. When the Global option is selected, the value from the Global Settings > Library Settings and Outputs > VoIP > SIP page is used.</p>

Wait Message Properties: Extract Variables

This page allows you to select the SIP variables you want to be extracted from the SIP message.

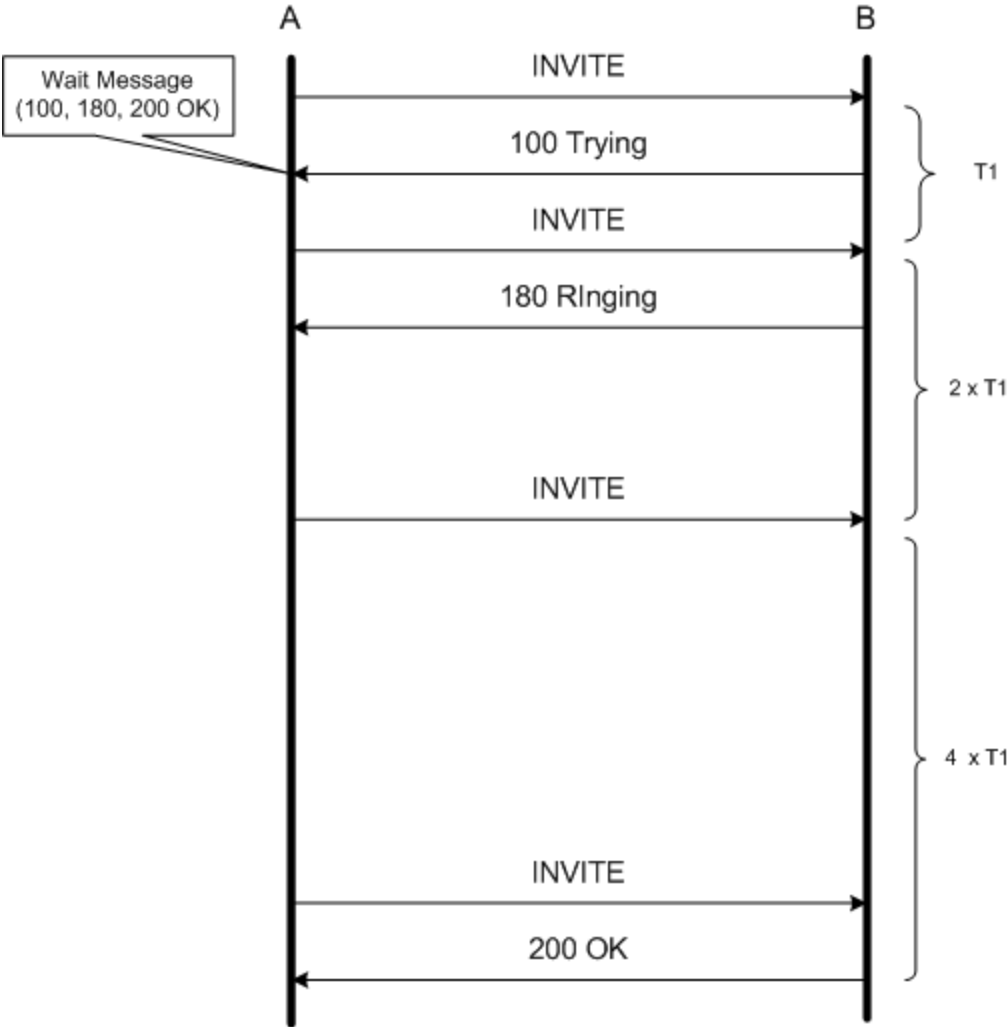
Wait Message Properties: Retransmission

The following table describes the outputs available for the Wait Response function:

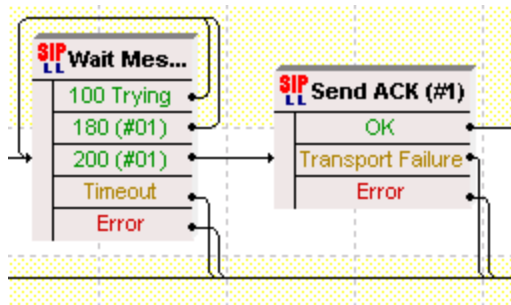
Parameter	Description
Retransmission Settings	<p>When a message comprised in a SIP transaction is matched, these settings define the conditions for ending or keeping transaction-level retransmissions of messages.</p> <hr/> <p> Note: For each script function that has retransmissions enabled and carries a SIP message subject to retransmissions, a retransmission rule is created. Each scenario channel maintains a list of known retransmission rules, one for every such script function that started retransmissions.</p>

Clean matched retransmission rule	If selected, the matched retransmission rule is stopped.
Keep selected retransmission rules active (and clean the other)	If selected, from all known retransmission rules displayed in the Started in Function drop-down, only the selected rule(s) remain(s) active.
From Template	Because the Wait Message function matches multiple message, the drop-down enables you to specify the list of retransmission rules kept active for each of the expected message templates.

For example, considering the messages flow shown in the image below:



whereby an INVITE message is retransmitted repeatedly by endpoint A at increasing intervals, and endpoint B responds with the successive 100 Trying, 180 Ringing and the 200 OK messages. For added flexibility, all response messages can be received using a single Wait Message script function, instead of multiple Wait Response ones, as shown in the image below:



The flow above corresponds to the following script functions settings:

- For the 100 Trying template, the **Keep selected retransmission rule active** option is selected and the **Send Invite** rule is enabled.
- For the 180 Ringing template, the **Keep selected retransmission rule active** option is selected and the **Send Invite** rule is enabled.
- For the 200 OK template, the **Keep selected retransmission rule active** option is selected, without any retransmission rule checked.

Wait Message Properties: Outputs

The following table describes the outputs available for Wait Message function:

Output Name	Description
The outputs corresponding to message template (max 10)	If the received SIP message matches one template, the corresponding output is enabled. The default resolution for this output is SUCCESS.
Timeout	This output is enabled if the received SIP message does not match any of the message templates in the specified timeout. The default resolution for this output is WARNING.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

Note: Any of the outputs corresponding to message template availability in this dialog depends on how the function options are configured.

Retransmit Last Message

Re-sends the last transmitted SIP Message.


Retransmit Last Message: Outputs

For more information, see [Send Request Properties: Outputs](#).

Extract Variables

Performs the extraction into one or more variables from a SIP message, or from parts of a SIP message.

Variable extraction is based on defining the extraction rules, as described in [Extraction Rules Definition](#). Existing variable extraction rules can be exported to the Resource Pool of the VoIP plug-in for later use; reuse of extraction rules from the Resource Pool is done by importing.

 **Note:** Because extraction of variables is done by a SIP parser software component, extraction rules definition must be compliant with the SIP syntax requirements.

Extract Variables: Parameters





Displays a list of variables selected for extraction and enables you to define the variable-specific extraction rules using a three-step wizard.

Variable Operations

Before defining the extraction rules, create a variable by clicking  and define it as follows:

- A Temporary variable – can be used only in the body of this function.
- A Scenario variable – is a user-defined, local variable.



In addition to creating a variable, further operations that can be performed on variables are as follows:

- Editing a variable, done by clicking .
- Deleting a variable, done by clicking the .
- Reordering the variable sequence, done by clicking  and .

Extraction Rules Definition

Extraction rules are defined using a wizard that comprises the following steps:

1. When and Where: Specifies a condition upon which extraction occurs and enables you to specify the SIP message from which the variable is extracted.
2. What: Defines the variable extraction scope, that is, if the variable is extracted from the entire message, or from parts of it.
3. Refine: Defines further processing operations that can be performed on the extracted variable string.

After a variable extraction rule has been defined, you can export it by clicking . To import an existing rule, click .

The following is a description of the options available for each extraction step listed above.

When and Where Step Parameters


The following table lists the parameters available for the When and Where step:

Area	Option	Description
When	Search only when the following expression is true	Selecting this option enables you to type in a conditional expression that determines whether variable extraction is done or not. For a description of the accepted elements and syntax, see The Expression Evaluator Syntax in Appendix B.
Where	Search in this protocol messages	Protocol type of messages to be searched.
	Search in last transmitted message	Select this option to perform the variable extraction from the last transmitted message.
	Search in last received and matched message	Select this option to perform the variable extraction from the last received and matched message.
	Search in variable	Select this option to perform the variable extraction from another variable that is specified in the adjacent drop-down control.

What Step Parameters






The following table lists the parameters available for the What step:

Parameter	Description
Entire message	Select this option to extract the variable from the entire message.
First Line	Select this option to extract the variable from the first line, or from parts of it, both for request and response messages. The following options are available: <ul style="list-style-type: none"> • Entire First Line • Request Line - Method • Request Line - Request-URI • Request Line - SIP Version • Status Line - SIP Version • Status Line - Status-Code • Status Line - Reason Phrase. Keep last CR/LF: Keeps the last Carriage Return/Line Feed character extracted into the variable.

Header	<p>Select this option to extract the variable from one or multiple occurrences of a message header. When choosing this option, the following options become available:</p> <ul style="list-style-type: none"> • Header type: Selects the header type from which the variable is extracted. • Compact form: After you have selected a header type, this field is automatically filled in with the header's compact form. • All occurrences from #: Sets the range of extracted occurrences of the chosen header type. If the occurrence count is greater than 1, multiple message lines corresponding to the multiple header rows are processed and extracted into the variable. <p>Extract options:</p> <ul style="list-style-type: none"> • The whole header value: Extracts the whole line, except for the header name. • Extract also the header name: Extracts the whole line, including the header name. • Header value without parameters: Extracts the header value, without any parameters. This option is available only for occurrence counts equal to 1. • Value of parameter named: Extracts the value of the named parameter. This option is available only for occurrence counts equal to 1. • Extract headers in reverse order: When the resulting occurrences count is higher than 1, choosing this option processes the occurrences in reverse order, starting from the last up to the first. • Keep last CR/LF: Keeps the last Carriage Return/Line Feed character extracted into the variable. <p>For example, when extracting multiple whole SIP headers into a variable and subsequently parsing that variable in order to perform another extraction, selecting this option ensures that the SIP parser finds valid, CR/LF-terminated lines, as specified by the SIP grammar.</p>
Message body	<ul style="list-style-type: none"> • Entire Message Body: The entire message body is extracted. <hr/> <p> Note: SIP messages may be in multiple parts. For information on multipart SIP messages, see Support for Multipart SIP Messages.</p> <hr/> <ul style="list-style-type: none"> • Only Content-Type: Only the selected type of content is extracted. <p>Extract options:</p> <ul style="list-style-type: none"> • The whole value: Extracts the value. • Value of object with name: Extracts the value of the named parameter. This option is available only for occurrence counts equal to 1. • All occurrences from #: Sets the range of extracted occurrences of the chosen header type. If the occurrence count is greater than 1, multiple message lines corresponding to the multiple header rows are processed and extracted into the variable.

Refine Step Parameters

The following table lists the parameters available for the Refine step:

Parameter	
Extract substring	
Use delimiters	<p>Extracts a substring delimited by a start (Begins after parameter) and an end (Ends before parameter) string. Both the start and end strings are user-specified. The Occurrence field defines the occurrence of the start and end strings.</p> <p>Example: Considering that your variable is a string comprising several comma-separated values, you can choose:</p> <ul style="list-style-type: none"> the ';' character and an occurrence count of '2' for the start string the ';' character and an occurrence count of '3' for the end string <p>The extracted substring would be that located between the 2nd and the 3rd occurrence of the ';' character.</p>
Use position	<p>Extracts a substring delimited by a start (Extract from position parameter) and an end position (to position parameter).</p>
Find & Replace	<p>Enables you to perform replacement operations on the resulting variable using Find & Replace rules. A Find & Replace rule is created by clicking , which opens a dialog box allowing you to specify the substring to search for, the replacement substring, and the occurrences for which to perform the replacement.</p> <p>Further operations that can be performed on Find & Replace rules are as follows:</p> <ol style="list-style-type: none"> Click  to edit a rule. Click  to delete a rule. Click  and  to reorder the rules.

Extract Variables: Outputs

The following table describes the outputs available for the Extract Variables function:

Output Name	Description
Found	The function completed successfully. The default resolution for this output is SUCCESS.
Not Found	The default resolution for this output is WARNING.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

MSRP Send AUTH

This function performs authentication for a MSRP endpoint against all relays specified in the VoIPSIP Peer activity's MSRP tab. If the authentication process succeeds, the returned response code is 200 OK

and the function exits on the OK output. If the authentication process does not complete successfully and a response is received other than 200 OK, if the received response is matched by any of the additional responses configured in the Rx Response tab, the function exits on the corresponding output, otherwise it exits on the Error output.

MSRP Send Auth: Tx Request Parameters

The following table describes the Send Auth Tx Request parameters:

Name	Description
Header name, Header value	Specifies the MSRP headers that are sent in the authentication request and enables you to configure them. The From-Path and To-Path headers are always contained in requests and automatically are populated with VoIPSIP activity level settings (<AUTO>). Other headers (Expires, Authorization) can be included optionally and their value can be configured using automatic settings (<AUTO>) configured at the VoIPSIP activity level, or numeric values.
Override SIP authentication	If selected, this enables you to define other authentication settings (user name, password) than those specified in the SIP configuration page of the VoIPSIP activity.

MSRP Send Auth: Rx Response Parameters

The following table describes the Send Auth Rx Response parameters:

Name	Description
Response to be matched	Specifies the expected response from authentication requests (200 OK is default), or any other user-defined response. For each expected response message, a new script function output is created. If the received response matches one of the specified responses, the script function exits on the output that corresponds to that response. If the response does not match any specified response, the function exists on the Error output. To create a new expected response, click Add , delete existing ones by clicking Delete . For example, assuming you are expecting a 400 Bad request response during the authentication process, you can choose to match this response code by creating a 400 Bad request entry.

MSRP Send Auth: Output Settings

The following table describes the default outputs available for the Multimedia Session function. Note that for each additional configured response, a new output is created.

Output Name	Description
-------------	-------------

OK	The function completed successfully. The default resolution for this output is SUCCESS.
Timeout	The function timed out while waiting for an authentication response. The default resolution of this output is timeout.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

MSRP Session

This function establishes an MSRP session for the simultaneous sending and receiving of text messages or files.

MSRP Session: Content Parameters


The following table describes the MSRP Session Content parameters:

Name	Description
Send MSRP Text Messages	<p>Specifies a text to be sent across the MSRP session.</p> <p>Send MSRP text messages: If selected, the specified text is sent.</p> <p>Send this message: Specifies a number of times the message is sent.</p> <p>Send each line in a separate message: If selected, each line (including the CRLF character) is sent as a separate message.</p>
Send files through MSRP	<p>When the Send Files through MSRP option is selected, these options specify the file to be sent across the MSRP session.</p> <ul style="list-style-type: none"> Send synthetic file of size: A synthetically created file of the specified size is sent. Send custom file: A real, user-specified file with a maximum size of 20 MB is sent. File negotiated in SDP: The transmitted file is one of the files defined in the VoIPSIP activity's MSRP configuration page, and that is being negotiated by SDP, as stipulated by RFC5547 - A Session Description Protocol (SDP) Offer/Answer Mechanism to Enable File Transfer. File # from activity settings: The transmitted file is one specified in the VoIPSIP activity's MSRP configuration page and identified by its table index.

MSRP Session: Settings


The following table describes the MSRP Session Settings parameters:

Name	Description
------	-------------

Session control	<p>Specifies how the MSRP session is terminated.</p> <ul style="list-style-type: none"> • Finish the session when: This specifies an aggregated criterion (AND logical operator) for terminating the session. • Finish the session when Tx is over: The MSRP session is terminated when the script function completes sending content. • Finish the session when at least x messages have been received: If selected, the MSRP session is terminated when a specified number of messages have been received. • Finish the session when file transfer Rx is over: If selected, the MSRP session is terminated when an incoming file transfer traffic is completed. • Finish the session at the end of the call: If selected, the MSRP session is only terminated at the end of the SIP session.
Tx control	<p>Specifies how MSRP traffic is handled.</p> <ul style="list-style-type: none"> • Max Tx chunk size: If selected, the MSRP content is 'chunked' into portions of the specified size. The maximum value for the chunk size is 1 MB. • Delay between text messages: If selected, the MSRP text content is send with delays of the specified size between text messages.
Non-blocking execution	<p>If selected, the MSRP Session script function starts executing and then returns control to the Scenario Editor, while continuing to execute in the background.</p> <hr/> <p> Note: When non-blocking execution is used, the script function's output status can be determined using the MSRP Control script function.</p> <hr/>

MSRP Session: Tx Requests Parameters

The following table describes the MSRP Session Tx Requests parameters:

Name	Description
Send request structure	<p>Specifies the structure of an MSRP Send request, as defined by RFC 4975, enabling you to select and configure MSRP header values.</p> <p>The From-Path, To-Path, and Message-ID headers are always contained in requests and are automatically populated with VoIPSIP activity level settings (<AUTO> setting). Other headers can be included optionally and their values can be configured using automatic activity level settings (<AUTO>), predefined values, or custom values.</p> <p>When clicking Add Custom header, one or more custom headers are added inside MSRP message, placed between the Failure-Report and the Content-ID header lines.</p> <hr/> <p> Note: IxLoad test scenario variables can be used as header values.</p> <hr/>

MSRP Session: Tx Responses Parameters

The following table describes the MSRP Session Tx Response parameters:

Name	Description
Response	<p>Enables you to configure the response for received MSRP requests.</p> <ul style="list-style-type: none"> Send automatic responses to all MSRP requests: If selected, responses are constructed automatically. Define custom response: If selected, the configured response (both response code and text comments can be configured) is sent for all received MSRP requests.

MSRP Session: CPIM Parameters

This tab adds support for the Message/CPIM MIME content type in MSRP (RFC 3862 and RFC 4975) and for the Instant Message Disposition Notifications (IMDN) mechanism (RFC 5438).

The following table describes the MSRP Session CPIM parameters:

Name	Description
Use CPIM encapsulation	<p>If selected, all MSRP SEND requests initiated by the current MSRP Session script function use a CPIM encapsulation.</p> <p>The CPIM 'To' and 'From' headers will have the same value as the \$VOIP_MSRP_CPIM_To and \$VOIP_MSRP_CPIM_From IxLoad variables.</p>
CPIM custom headers	<p>If selected, one or more custom headers are added inside the CPIM headers.</p> <p>The custom headers may contain VoIP variables.</p>
Add disposition notification (IMDN)	<p>If selected, IMDN notifications are requested for the received MSRP SEND requests. This mechanism is used together with CPIM.</p> <p>The Positive delivery, Negative Delivery, Display notification options specify the requested notification types.</p> <p>In the IxLoad implementation, an emulated VOIPSIP Peer activity can respond only with a positive delivery or a display notification.</p>
Delay between notifications (ms)	<p>The time interval that specifies the delay between the positive delivery of MSRP messages and the display notifications (if both were requested).</p>

The following example illustrates an MSRP SEND request with CPIM and IMDN enabled:

```
MSRP 00000419da SEND
To-Path: msrp://ndc101cpm.npc.mobilephone.net:32398/
n01s00i3t554BDFA5+121714;tcp
From-Path: msrp://ndc101cpm.npc.mobilephone.net:1024/
db7b02;tcp
Message-ID: eblc000400
Byte-Range: 1-303/303
Success-Report: no
Failure-Report: yes
Content-ID: <b840000004@ndc101cpm.npc.mobilephone.net>
Content-Description: A simple text message
Content-Type: message/cpim
```

```

From: <sip:+18800300000@one.net>
To: <tel:+18800305000>
DateTime: 2015-05-12T09:28:58Z
NS: imdn <urn:ietf:params:imdn>
imdn.Message-ID: 374b18a22290009433
imdn.Disposition-Notification: positive-delivery,
display
Content-Type: text/plain
"Let's communicate"
-----00000419da$

```

MSRP Session: Output Parameters

The following table describes the outputs available for the MSRP Session function:

Output Name	Description
OK	The function completed successfully. The default resolution for this output is SUCCESS.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

MSRP Control

The MSRP Control function has the purpose of probing or controlling one (the previously executed) MSRP Session script function that had non-blocking behavior enabled.

Note: When running in non-blocking execution mode, script functions perform their tasks in background, allowing simultaneous handling of other signaling and media functions. A MSRP Session script function running in non-blocking mode always exits on the SUCCESS output, while the 'true' result of the function execution is obtained by evaluating the output of the MSRP Control function.

If blocking execution is enabled for a previously executed MSRP Session function, the output of the MSRP Control function indicates whether the function execution completed (MSRP Not Running output) or not (MSRP Running output).

MSRP Control: Control Action Parameters

The following table describes the MSRP Control Action parameters:

Name	Description
------	-------------

Delay Before Execution	<p>Delays the function execution by a duration that can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – Fixed Delay Before Execution duration. It can be used for synchronization reasons. It can be a value in milliseconds (ms), or a formula. • Random Between Expressions – Random Delay Before Execution duration value in the specified interval. It can be used to simulate real-life conditions. <p>The default is Static Expression, 100 ms.</p>
Control Actions	<p>Specifies whether the function only probes the status of an executing MSRP Session function (Check for MSRP completion option), or whether it actually terminates it (Terminate MSRP option).</p>

MSRP Control: Output Parameters

The following table describes the outputs available for the MSRP Control function:

Output Name	Description
MSRP Running	The previous MSRP function is still executing. The default resolution for this output is SUCCESS.
MSRP Not Running	The previous MSRP function has completed execution. The default resolution for this output is SUCCESS.
Timeout	The function execution has generated a timeout. The default resolution for this output is WARNING.
Error	The function has returned an internal error. The default resolution for this output is FAILED.


VoIP Skinny Functions Library

The VoIP Skinny Test Library includes the following script functions:

- [RegisterClient](#)
- [UnregisterClient](#)
- [OffHook](#)
- [OnHook](#)
- [NewCall](#)
- [EndCall](#)
- [MakeCall](#)
- [WaitCall](#)
- [AnswerCall](#)
- [DialDigits](#)
- [WaitDigits](#)
- [HoldCall](#)
- [RetrieveCall](#)
- [Setup XFER](#)
- [Complete XFER](#)
- [Transfer](#)
- [ForwardAllCalls](#)
- [ParkCall](#)
- [GetCallInfo](#)
- [MeetMe](#)
- [RemoveLast ConferenceParty](#)
- [SendStimulus](#)
- [SendSoftkey](#)
- [IsSoftKeyAvailable](#)
- [WaitForEvent](#)

RegisterClient

Registers the Skinny client with a specified Cisco CallManager. If the client is already registered with the CallManager, this function is skipped.

 **Note:** For tests comprising several loops, the registration state is maintained across loops in the test.

The RegisterClient script function parameters are given in the following table:

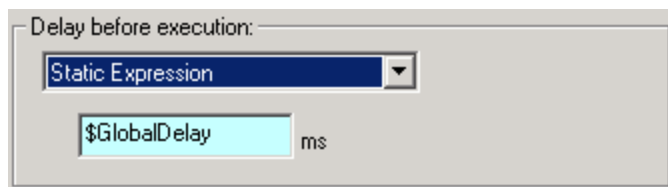
Properties	Description
------------	-------------

Parameters	
Device Settings	<p>Specifies the station settings for registration with the CallManager:</p> <ul style="list-style-type: none"> • Use Dial Plan: The registration name and type are those defined in the dial plan. • Registration Name: An alternative registration name and type are provided.
Register to	<p>Enables you to specify a CallManager address and port overriding the settings from the Skinny Settings page:</p> <ul style="list-style-type: none"> • Server Address: The CallManager IP address • Server Port: The CallManager port
Type	<p>The emulated station type, which can be one of the following:</p> <ul style="list-style-type: none"> • Device Station 30 SP+ • Device Station 12 SP+ • Device Station 12 SP • Device Station 30 VIP • Device Station Telecaster (Cisco IP Phone 7910) • Device Station TelecasterMgr (Cisco IP Phone 7960) • Device Station TelecasterBus (Cisco IP Phone 7940) • Device Station Polycom (Cisco IP Phone 7935) • Device Cisco IP Phone 7902 • Device Station VGC (Cisco VGC Phone)
Delay before execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from the VoIP Plug-In Resource Pool Constants category.</p> <p>The default is Static Expression, 0 ms.</p>
Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be a Global constant, a value in ms, or a timeout from the VoIP Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Advanced	<p>Permits you to specify a registration sequence with the CallManager. The registration sequence is given below:</p>

Message	Direction
StationKeepAlive	To CCM
StationKeepAliveAck	From CCM
StationAlarm	To CCM
StationRegister	To CCM
StationIpPort	To CCM
StationRegisterAck	From CCM
*StationHeadsetStatusMessage	To CCM
StationCapabilitiesReq	From CCM
StationCapabilitiesRes	To CCM
StationTemplateReq	To CCM
StationTemplateRes	From CCM
*StationSoftKeyTemplateReq	To CCM
*StationSoftKeyTemplateRes	From CCM
*StationSoftKeySetReq	To CCM
*StationSoftKeySetRes	From CCM
StationLineStatReq	To CCM
*StationSelectSoftKey	From CCM
*StationDisplayPromptStatus	From CCM
StationLineStat	From CCM
StationSpeedDialStatReq	To CCM
StationSpeedDialStat	From CCM
*StationRegisterAvailableLinesMessages	To CCM
StationDateTimeReq	To CCM
StationDefineDateTime	From CCM

<p>Custom Skinny Registration Sequence:</p>	<p>If selected, a custom sequence can be specified by clicking the message entries that are to be included in the custom sequence. When this option is not selected, the default sequence used ignores the StationKeepAliveAck and StationDisplayPromptStatus messages.</p> <hr/> <p>Note: Messages marked with an asterisk (*) are supported by the Cisco IP phone 7960 or 7940 style devices and are not used by 12- and 30- set emulations.</p> <hr/> <p>Note: If a message is not applicable for the device, it is skipped together with the pertaining response, if one is expected.</p>
<p>Output Settings</p>	
<p>Ok</p>	<p>The client registration was done successfully. The default resolution for this output is SUCCESS.</p>
<p>Error</p>	<p>The function has returned an error. The default resolution for this output is FAILED.</p>

Fields highlighted in blue (as shown below) in the script function configuration tabs indicate that expressions using scenario variables and numerical values are accepted as input in these fields.



UnregisterClient


Unregisters a Skinny client from the Cisco CallManager. The UnregisterClient script function parameters are described in the following table:


Properties	Description
<p>Parameters</p>	
<p>Delay Before Execution</p>	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from the VoIP Plug-In Resource Pool Constants category.</p> <p>The default is Static Expression, 0 ms.</p>

Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be a Global constant, a value in ms, or a timeout from the VoIP Constants pool. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as, for example, Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	The client deregistration completed successfully. The default resolution for this output is SUCCESS.
Error	The function has returned an error. The default resolution for this output is FAILED.

OffHook

Notifies the Cisco CallManager that the station is in an off-hook state. The Off-Hook script function parameters are described in the following table:

Properties	Description
Parameters	
OffHook HCall	<p>A new automatically-generated, unique call reference with a CallHandle#%d format. For every Skinny function that creates a new call reference (HCall), such as OffHook, the call handle number is increased.</p> <p>For example, considering that the first OffHook function that has been added to the scenario has an HCall equal to CallHandle#1, the second OffHook function, or any other function that creates a new call reference, added to the scenario, has a CallHandle#2 call reference.</p> <hr/> <p> Note: Call handles cannot be changed, they are read-only.</p>
Line No	The station line number.
Ok	The client deregistration completed successfully. The default resolution for this output is SUCCESS.
Error	The function has returned an error. The default resolution for this output is FAILED.

Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from the VoIP Plug-In Resource Pool.</p> <p>The default is Static Expression, 0 ms.</p>
Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p> <hr/> <p> Note: If the function is made up of more than one Skinny message, then the timeout is set for the whole duration of the command.</p>
Output Settings	
Ok	The notification was sent successfully. The default resolution for this output is SUCCESS.
Error	The function has returned an error following an inexistent call handle, inexistent line, or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

OnHook

Notifies the Cisco CallManager that the station is in an on-hook state and disconnects all active calls. The OnHook script function parameters are described in the following table:

Properties	Description
Parameters	
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from the VoIP Plug-In Resource Pool.</p> <p>The default is Static Expression, 0 ms.</p>

Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	The notification was sent successfully. The default resolution for this output is SUCCESS.
Timeout	A timeout occurred while sending the notification. The default resolution for this output is WARNING.
Error	The function has returned an error. The default resolution for this output is FAILED.

NewCall

Sends a SoftKeyEvent message to the Cisco CallManager, requesting a dial tone. The NewCall script function parameters are described in the following table:

Properties	Description
Parameters	
NewCall HCall	A new automatically-generated, unique call reference with a CallHandle#%d format.
Line No	The station line number.
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from the VoIP Plug-In Resource Pool.</p> <p>The default is Static Expression, 0 ms.</p>

Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	The notification was sent successfully. The default resolution for this output is SUCCESS.
Error	The function has returned an error following an inexistent call handle, inexistent line, or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

EndCall

Sends a SoftKeyEvent message to the Cisco CallManager, requesting a specific call completion. The EndCall script function parameters are described in the following table:

Properties	Description
Parameters	
Call Handle	A reference to an established call from those that have been created in the scenario by Skinny functions such as OffHook or NewCall. If the AUTO option is selected, the call handle corresponds to the active call.
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from the VoIP Plug-In Resource Pool Constants category.</p> <p>The default is Static Expression, 0 ms.</p>

Wait Other Party To Disconnect	<p>If selected, the station waits for a Callstate message with a TsOnHook value. If it does not receive the message within an user-defined timeout, it sends an Endcall. The timeout value can be specified as either of the following:</p> <ul style="list-style-type: none"> • Static Expression: The timeout is specified by a static expression. • Random between (from [value] to [value]): The timeout is a random value within a specified interval. • Wait for other party: The timeout value is a constant from the VoIP Plug-in Resource Pool Constants category.
Output Settings	
Ok	The call ended successfully. The default resolution for this output is SUCCESS.
Error	The function has returned an error following an inexistent call handle or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

MakeCall

Originates a call by dialing the phone number and performing the call establishment. The MakeCall script function parameters are described in the following table:

Properties	Description
Parameters	
NewCall HCall	A new automatically-generated, unique call reference with a CallHandle#%d format.
Line No	The station line number.
Destination	<p>Specifies the call destination phone number as either of the following:</p> <ul style="list-style-type: none"> • Use Dial Plan Settings: The call destination is given by the Dial Plan page settings. • Phone: The call destination is given by a user-defined string representing a valid phone number. • Last Parked Call: The call destination is the last parked call. • Last Dialed Number: Instructs the CallManager to use the last dialed number.
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from the VoIP Plug-In Resource Pool Constants category.</p> <p>The default is Static Expression, 0 ms.</p>

Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	A call was successfully established. The default resolution for this output is SUCCESS.
Timeout	A timeout occurred while sending the notification. The default resolution for this output is WARNING.
Disconnected	On the established connection, the remote party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.
Error	The function has returned an error following an inexistent call handle, inexistent line, or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

WaitCall

Waits for an incoming call. The WaitCall script function parameters are described in the following table:

Properties	Description
Parameters	
HCall	A new automatically-generated, unique call reference with a CallHandle#%d format.
Line No	The line number on which the call is to be established. If the ANY option is selected, the station listens on all available lines.
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from the VoIP Plug-In Resource Pool Constants category.</p> <p>The default is Static Expression, 0 ms.</p>

Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	A call was successfully established. The default resolution for this output is SUCCESS.
Timeout	A timeout occurred while sending the notification. The default resolution for this output is WARNING.
Error	The function has returned an error following an inexistent call handle, inexistent line, or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

AnswerCall

This function, which is used conjointly with the previous WaitCall function, answers an incoming call by going off-hook and performing the call establishment. The AnswerCall script function parameters are described in the following table:

Properties	Description
Parameters	
Call Handle	A reference to an established call. If the AUTO option is selected, the call handle corresponds to the active call.
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from the VoIP Plug-In Resource Pool Constants category.</p> <p>The default is Static Expression, 0 ms.</p>

Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	A call was successfully answered. The default resolution for this output is SUCCESS.
Timeout	A timeout occurred while sending the notification. The default resolution for this output is WARNING.
Disconnected	On the established connection, the other party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.
Error	The function has returned an error following an inexistent call handle, inexistent line, or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

DialDigits

Dials the specified DTMF digits. The DialDigits script function parameters are described in the following table:

Properties	Description
Parameters	
Dial String	<p>Specifies the dialed digits as either of the following:</p> <ul style="list-style-type: none"> • Use Dial Plan Settings: The dial string is the destination specified in the Dial Plan page. • Phone: A user-defined string representing a valid phone. • Last Parked Call: The dial string corresponds to the last parked call.

Delay Between Digits	<p>The inter-digit delay, which can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed delay specified as a value in milliseconds (ms) or as a delay constant from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random delay duration value in the specified interval. • A user-defined delay specified as a constant from the VoIP Plug-In Resource Pool Constants. <p>The default is Static Expression, 100 ms.</p>
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from the VoIP Plug-In Resource Pool Constants category.</p> <p>The default is Static Expression, 0 ms.</p>
Output Settings	
Ok	Digits were successfully dialed. The default resolution for this output is SUCCESS.
Disconnected	On the established connection, the other party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.
Error	The function has returned an error following an inexistent call handle, inexistent line, or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

WaitDigits

Waits for a StationKeypadMessage sent by the Cisco CallManager to the station and detects a DTMF digits sequence. The WaitDigits script function parameters are described in the following table:

Properties	Description
Parameters	
DTMF Detection Settings	<p>Specifies the detection mode settings, as follows:</p> <ul style="list-style-type: none"> • Detect continuously for – Detects all digits arrived within the specified period of time (Default = 1000 s). • Detect exactly <x> DTMFs – Detects the specified number of digits (Default = 6). • Use Talk Time – Detects DTMFs for the duration of the TalkTime test configuration parameter. • Detect DTMF sequence – Detects an expected sequence, user-defined or specified by selecting a DTMF Sequence Pool entry.

Terminate conditions	<p>Maximum Delay between DTMFs: The maximum amount of time, in milliseconds (ms), allowed between consecutive digits for a proper detection. After this period elapses, the function terminates. The range of values is 0 to 999999 ms. The default value is 500 ms.</p> <p>First DTMF Timeout: The time, in milliseconds (ms), allowed for receiving the first digit. After this period elapses, the function exits on the Timeout output. The range of values is 0 to 999999 ms. The default value is 400 ms.</p>
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from the VoIP Plug-In Resource Pool Constants category.</p> <p>The default is Static Expression, 0 ms.</p>
Output Settings	
Ok	The digits were successfully detected. The default resolution for this output is SUCCESS.
Timeout	The digits were not detected due a timeout. The default resolution for this output is WARNING.
Disconnected	On the established connection, the other party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.
Error	The function has returned an error following an inexistent call handle, inexistent line, or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

HoldCall

Performs a Hold operation on a specified call reference. The HoldCall script function parameters are described in the following table:

Properties	Description
Parameters	
Call Handle	A reference to the call to be put on hold. If the ANY option is selected, the currently active call is put on hold.
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from VoIP Plug-In Resource Pool Constants.</p> <p>The default is Static Expression, 0 ms.</p>

Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	The call was successfully put on hold. The default resolution for this output is SUCCESS.
Error	The function has returned an error following an inexistent call handle or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

RetrieveCall

Performs a Retrieve operation on a specified call reference. The RetrieveCall script function parameters are described in the following table:

Properties	Description
Parameters	
Call Handle	A reference to the call to be retrieved from the hold state. If the AUTO option is selected, the call handle corresponds to the last active call.
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from VoIP Plug-In Resource Pool Constants.</p> <p>The default is Static Expression, 0 ms.</p>

Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	The call was successfully retrieved from the hold state. The default resolution for this output is SUCCESS.
Error	The function has returned an error following an inexistent call handle or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

Setup XFER

Initiates a Transfer or a Conference procedure, without effectively completing the procedure. The actual transfer is performed using the complementary Complete XFER script function. The Setup XFER script function parameters are described in the following table:

Properties	Description
Parameters	
NewCall Hcall	A new automatically-generated, unique call reference with a CallHandle#%d format.
XFer Option	TRANSFER CALL or CONFERENCE CALL.
XFer CallHandle	A reference to the call on which to perform the action.
Destination	<p>Defines the call transfer destination as either of the following:</p> <ul style="list-style-type: none"> • Use Dial Plan Settings: The transfer destination is given by the Dial Plan page settings. • Phone: A user-defined string representing a valid phone number. • Last Parked Call: The transfer destination is the last parked call. • Last Dialed Number: Instructs the CallManager to use the last dialed number.

Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from VoIP Plug-In Resource Pool Constants.</p> <p>The default is Static Expression, 0 ms.</p>
Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	The transfer/conference procedure was successfully initiated, that is, a call was established with the third party. The default resolution for this output is SUCCESS.
Timeout	A timeout occurred while sending the notification. The default resolution for this output is WARNING.
Disconnected	On the established connection, the remote party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.
Error	The function has returned an error following an inexistent call handle or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

Complete XFER

Completes a Transfer or Conference procedure that was previously initiated using the Setup XFER function. The CompleteXFER script function parameters are described in the following table:

Properties	Description
Parameters	
Call Handle	A reference to the call on which to perform the action.
XFer Option	TRANSFER CALL or CONFERENCE CALL.

Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from VoIP Plug-In Resource Pool Constants.</p> <p>The default is Static Expression, 0 ms.</p>
Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	<p>The transfer/conference procedure was successfully completed. For a transfer, the initial call is closed (for transfer) and the other parties should be connected. For a conference procedure, the call is connected into a conference. The default resolution for this output is SUCCESS.</p>
Timeout	<p>A timeout occurred while sending the notification. The default resolution for this output is WARNING.</p>
Disconnected	<p>On the established connection, the remote party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.</p>
Error	<p>The function has returned an error following an inexistent call handle or erroneous parameters evaluation condition. The default resolution for this output is FAILED.</p>

Transfer

This function, which is a combination of the Setup XFER and the Complete XFER functions, transfer the call to another party. The Transfer script function parameters are described in the following table:

Properties	Description
Parameters	
NewCall Hcall	A new automatically-generated, unique call reference with a CallHandle#%d format.


Destination	<p>Defines the call transfer destination as either of the following:</p> <ul style="list-style-type: none"> • Use Dial Plan Settings: The transfer destination is given by the Dial Plan page settings. • Phone: A user-defined string representing a valid phone number. • Last Parked Call: The transfer destination is the last parked call. • Last Dialed Number: Instructs the CallManager to use the last dialed number.
Transfer Mode	<p>Defines the transfer mode, with or without consultation of the party to which the call is transferred:</p> <ul style="list-style-type: none"> • Blind Transfer: A new call is initiated without consulting the party to which the call is transferred. The call is established without waiting for the third party to answer, and the callee should not answer before transfer is completed. If the third party is an IxLoad emulated Skinny phone executing an AnswerCall script function, the use of a delay before the function execution is recommended. • Consultative transfer with duration: Before transferring the call, the remote party is being consulted for a specified or random period of time.
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from VoIP Plug-In Resource Pool Constants.</p> <p>The default is Static Expression, 0 ms.</p>
Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	The transfer procedure was successfully performed. The default resolution for this output is SUCCESS.
Timeout	A timeout occurred while sending the notification. The default resolution for this output is WARNING.
Disconnected	On the established connection, the remote party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.

Error	The function has returned an error following an inexistent call handle or erroneous parameters evaluation condition. The default resolution for this output is FAILED.
-------	--

ForwardAllCalls

Sends a SoftKeyEvent message to the Cisco CallManager, requesting that all incoming calls be forwarded. The ForwardAllCalls script function parameters are described in the following table:


Properties	Description
Parameters	
Destination	<p>Defines the call transfer destination as either of the following:</p> <ul style="list-style-type: none"> • Use Dial Plan Settings: The transfer destination is given by the Dial Plan page settings. • Phone: A user-defined string representing a valid phone number. • Last Parked Call: The transfer destination is the last parked call. • Last Dialed Number: Instructs the CallManager to use the last dialed number.
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from VoIP Plug-In Resource Pool Constants.</p> <p>The default is Static Expression, 0 ms.</p>
Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	The forward configuration was successfully performed. The default resolution for this output is SUCCESS.
Error	The function has returned an error following an inexistent call handle or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

 **Note:** Since during the first execution, the ForwardAllCalls script function configures the forwarding functionality on the CCM, it needs to be executed twice to effectively forwards the call.

ParkCall

Parks a specified call. The ParkCall script function parameters are described in the following table:

Properties	Description
Parameters	
Call Handle	A reference to an established call. If the ANY option is selected, the active call is parked.
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from VoIP Plug-In Resource Pool Constants.</p> <p>The default is Static Expression, 0 ms.</p>
Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	The call was successfully parked. The default resolution for this output is SUCCESS.
Error	The function has returned an error following an inexistent call handle or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

 **Note:** The call parking and call pick up operations need to be executed on the same test scenario channel.

GetCallInfo

Retrieves the call information (CallInfo and CallState) into the Skinny variables supported by the IxLoad VoIP Plug-In. The GetCallInfo script function parameters are described in the following table:

Properties	Description
Parameters	
Call Handle	A reference to an established call.
Delay Before Execution	The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels. The delay value can be specified as a static expression, a random value, or as a predefined constant from VoIP Plug-In Resource Pool Constants. The default is Static Expression , 0 ms.
Output Settings	
Ok	The retrieval of call parameters into Skinny variables was completed successfully. The default resolution for this output is SUCCESS.
Error	The function has returned an error following an inexistent call handle or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

MeetMe

Sets up a MeetMe conference with the test script function parameters described in the following table. To set up the conference, a Meet-Me softkey message is sent to the Cisco CallManager and then the Meet-Me conference call number is dialed.

Stations that want to connect to the conference have to dial in to the conference using the MakeCall script function.

Properties	Description
Parameters	
NewCall HCall	A new automatically-generated, unique call reference with a CallHandle#%d format.
Line No	The station line number.

Destination	<p>Specifies the call destination phone number as either of the following:</p> <ul style="list-style-type: none"> • Use Dial Plan Settings: The call destination is given by the Dial Plan page settings. • Phone: The call destination is given by a user-defined string representing a valid phone number. • Last Parked Call: The call destination is the last parked call. • Last Dialed Number: Instructs the CallManager to use the last dialed number.
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from the VoIP Plug-In Resource Pool Constants category.</p> <p>The default is Static Expression, 0 ms.</p>
Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
Ok	The MeetMe-type conference call setup completed successfully. The default resolution for this output is SUCCESS.
Timeout	A timeout occurred while sending the notification. The default resolution for this output is WARNING.
Disconnected	On the established connection, the remote party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.
Error	The function has returned an error following an inexistent call handle, inexistent line, or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

RemoveLast ConferenceParty

Sends a RmLstC softkey to the Cisco CallManager that removes from a conference the party that has connected last. The RemoveLastConferenceParty script function parameters are described in the following table:

Properties	Description
Parameters	
Call Handle	A reference to an established call. If the AUTO option is selected, the call that last joined the conference is removed.
Delay Before Execution	The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels. The delay value can be specified as a static expression, a random value, or as a predefined constant from VoIP Plug-In Resource Pool Constants. The default is Static Expression , 0 ms.
Output Settings	
Ok	The RmLstC softkey was sent successfully. The default resolution for this output is SUCCESS.
Error	The function has returned an error following an inexistent call handle or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

SendStimulus

A Skinny station uses this message to inform the Cisco CallManager that a functional stimulus was pressed. The SendStimulus script function parameters are described in the following table:

Properties	Description
Parameters	
Line No	The station line on which the stimulus is transmitted.

Device Stimulus	<p>The stimulus that is transmitted to the CallManager, which can be either of the following:</p> <ul style="list-style-type: none"> • SsLastNumberRedial = 1, • SsSpeedDial = 2, • SsHold = 3, • SsTransfer = 4, • SsForwardAll = 5, • SsForwardBusy = 6, • SsForwardNoAnswer = 7, • SsDisplay = 8, • SsLine = 9, • SsT120Chat = 0xA, • SsT120Whiteboard = 0xB, • SsT120ApplicationSharing = 0xC, • SsT120FileTransfer = 0xD, • SsVideo = 0xE, • SsVoiceMail = 0xF, • SsAnswerRelease = 0x10, • SsAutoAnswer = 0x11, • SsSelect = 0x12, • SsPrivacy = 0x13, • SsServiceURL = 0x14, • SsMaliciousCall = 0x1B, • SsGenericAppB1 = 0x21, • SsGenericAppB2 = 0x22, • SsGenericAppB3 = 0x23, • SsGenericAppB4 = 0x24, • SsGenericAppB5 = 0x25 • SsMeetMeConference=0x7b, • SsConference=0x7d, • SsCallPark=0x7e • SsCallPickup=7f • SsGroupCallPickup=80
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from VoIP Plug-In Resource Pool Constants.</p> <p>The default is Static Expression, 0 ms.</p>

Output Settings	
<stimulus name>	This output, named after the stimulus selected in the Parameters page, indicates that the stimulus was successfully sent. For custom stimuli, the value must have the 'User defined #%' format.
Error	The function has returned an error following an inexistent line number or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

SendSoftkey

CP-7940/7960 stations use this message to inform the Cisco CallManager of a softkey event. The SendSoftKey script function parameters are described in the following table:

Properties	Description
Parameters	
Call Handle	A reference to an established call.
Delay Before Execution	The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels. The delay value can be specified as a static expression, a random value, or as a predefined constant from VoIP Plug-In Resource Pool Constants. The default is Static Expression , 0 ms.

SoftKey	<ul style="list-style-type: none"> • SkCFwdBusy • SkCFwdNoAnswer • SkBackSpace • SkEndCall • SkResume • SkAnswer • SkInfo • SkConfrn • SkPark • SkJoin • SkMeetMeConfrn • SkCallPickUp • SkGrpCallPickUp • SkRmLstC • SkSelect • SkDirTrFt • SkCongList • SkRedial • SkNewCall • SkHold • SkTrnsfer • SkCFwdAl
Output Settings	
<softkey name>	The output, named after the softkey selected in the Parameters page, indicates that the softkey was successfully sent.
Error	The function has returned an error following an inexistent call handle or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

 **Note:** This function is available only for devices that have completed registration as devices compatible with CP-7940/60.

IsSoftKeyAvailable

Verifies if a specified soft key is available on the Skinny station. The IsSoftKeyAvailable script function parameters are described in the following table:

Properties	Description
Parameters	
Softkey	The softkey value that is verified for availability.

Use	Specifies the active line or the call subject to verification.
Delay Before Execution	The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels. The delay value can be specified as a static expression, a random value, or as a predefined constant from VoIP Plug-In Resource Pool Constants. The default is Static Expression , 0 ms.
Output Settings	
<softkey name>	This output, named after the softkey selected in the Parameters page, indicates that the softkey is available.
Error	The softkey selected in the Parameters page is not available. The default resolution for this output is FAILED.

WaitForEvent

The Skinny Client searches the message queue for a specified message in a given scope and waits for it if the message is not present. The WaitForEvent script function parameters are described in the following table:

Properties	Description
Parameters	
Scope	Specifies the search domain as one of the following: <ul style="list-style-type: none"> • Call Handle: The message is waited for in the specified call. • Call State: The message is searched for in the message queue beginning with the specified state.
Generate New HCall	When selected, the receiving of the expected message generates a new call if the call ID is not associated with an existing call handle.

Skinny Message	<p>The Skinny message for which to wait. For messages that configure parameters with multiple values, the drop-down list below permits you to select a value. For example, when selecting the SendDtmfToneMessage in the Skinny Message list, the awaited value can be any DTMF or a specific DTMF, which can be selected from the list below:</p> <ul style="list-style-type: none"> • StationActivateCallPlaneMessage • StationBackSpaceReqMessage • StationTemplateMessage • StationCallInfoMessage • StationCall State Message <ul style="list-style-type: none"> ▪ Any ▪ TsIdle ▪ TsOffHook ▪ TsOnHook ▪ TsRingOut ▪ TsRingIn ▪ TsConnected ▪ TsBusy ▪ TsCongestion ▪ TsHold ▪ TsCallWaiting ▪ TsCallTransfer ▪ TsCallPark ▪ TsProceed ▪ TsCallRemoteMultiline ▪ TsInvalidNumber • StationCapabilitiesReqMessage • StationClearDisplay • StationClearNotifyMessage • StationClearPromptStatusMessage • StationCloseReceiveChannel • StationConfigStatMessage • StationConnectionStatisticsReq • StationDeactivateCallPlaneMessage • StationDefineTimeDate • StationDisplayNotifyMessage • StationDisplayPromptStatusMessage • StationDisplayTextMessage • StationForwardStatMessage • StationKeepAliveAckMessage • StationLineStatMessage • StationOpenReceiveChannel • StationRegisterAckMessage • StationRegisterRejectMessage • StationRegisterTokenAck
	<ul style="list-style-type: none"> • StationRegisterTokenReject • StationReset • StationSelectSoftKeysMessage • StationServerResMessage

Empty Message Queue	The message is deleted from the message queue.
Delay Before Execution	<p>The time to wait before the function starts. Introducing a delay is used for synchronization reasons, such as, for example, to synchronize functions on different scenario channels.</p> <p>The delay value can be specified as a static expression, a random value, or as a predefined constant from VoIP Plug-In Resource Pool Constants.</p> <p>The default is Static Expression, 0 ms.</p>
Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. The timeout value can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression – A fixed timeout duration, which can be Global constant, a value in ms, or a timeout from the VoIP Plug-in Resource Pool Constants category. • Random Between Expressions – A random timeout duration value in a specified interval. • Any of the timeout constants from the VoIP Plug-in Resource Pool Constants, such as for example Delay between digits or PHONE_WAIT_TIME. <p>The default is Static Expression, 20000 ms.</p>
Output Settings	
<message to wait for>	This output, named after the message selected in the Parameters page, indicates that the message was received. The default resolution for this output is SUCCESS.
Timeout	A timeout occurred while receiving the message. The default resolution for this output is WARNING.
Error	The specified message has not been received. The default resolution for this output is FAILED.

VoIP Media Functions Library

The VoIP Media functions are used to generate media streaming for calls established by using either VoIP protocol implementation.

The currently implemented Media functions are the following:

- [Generate DTMF](#)
- [Detect DTMF](#)
- [Generate MF](#)
- [Detect MF](#)
- [Generate Tone](#)
- [Wait for Tone](#)
- [Talk](#)
- [Listen](#)
- [Voice Session](#)
- [Multimedia Session](#)
- [MCPTT Send](#)
- [MCPTT Wait](#)
- [T.38 Fax Session](#)
- [Path Confirmation](#)
- [RTP Control](#)

Generate DTMF

Generates a specified DTMF sequence.

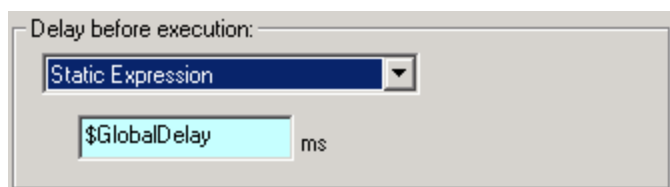
Generate DTMF: Parameters

The following table describes the Generate DTMF function parameters:

Name	Description
Delay Before Execution	<p>The time to wait before the function starts. It can be used for synchronization reasons (that is, to synchronize the Make Call function with the Receive Call one). It can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, Sleep 2000 in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) <p>The default is Static Expression, 20000 ms.</p>

DTMF Sequence	Selects a specific DTMF sequence item from those defined in the DTMF Sequence Pool Items or Local Sequence. The default is 12345.
DTMF Duration	The time (in ms) required for a single tone to be generated. The range of values is 0 to 999999 ms. The default value is 100 ms.
Inter DTMF Interval	The maximum amount of time (in ms) between two consecutive generated DTMFs. The range of values is 0 to 999999 ms. The default value is 200 ms.
DTMF Amplitude	The attenuation (in dB) of the DTMF tone. The minimum attenuation is 0 dB (no attenuation) and the maximum is -40 dB. The default value is -10 dB.

Fields highlighted blue (shown in the image below) in the script function configuration tabs indicate that expressions using scenario variables and numerical values are accepted as input in these fields.



Generate DTMF: Advanced Playback Settings

The following table describes the Generate DTMF function advanced playback settings:

Name	Description
Transmission Mode	<p>Generates a sequence of DTMFs. The sequence can have up to 31 tones and it can contain any combination of the standard tones, (that is, '1', '2', '3', '4', '5', '6', '7', '8', '9', '0', '#', '*', 'A', 'B', 'C', 'D.') The DTMF sequence can be transmitted In Band or Out of Band using a 2833 event payload format or the signaling layer and it can be played continuously for a number of times or for a specific period of time.</p> <ul style="list-style-type: none"> • In Band: Using RTP media streaming. • Out of Band – Using 2833 EVENT Payload Format: The RTP Payload format used for carrying dual-tone multi frequency (DTMF) digits and other line and trunk signals as events. • Out of Band – Using 2833 TONE Payload Format: Using 2833 TONE Payload Format–The RTP Payload format that can represent tones consisting of one or more frequencies. For details on RTP Payload, see RFC 2833 - RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals. • Use Global Settings: Use the settings from the Global Settings > Library Settings and Outputs > VoIP > RTP page.

Playback	<ul style="list-style-type: none"> • Play: Specifies the number of times to play. • Repeat Continuously for: Specify the period of time to play. The default value is 1000 seconds. • Use Talk Time (only for BHCA objective): Plays the DTMF for the duration of the Talk Time call parameter. • Use Global Settings: Use the settings from the Global Settings > Library Settings and Outputs > VoIP > RTP page.
----------	--

Generate DTMF: Outputs

The following table describes the outputs available for the Generate DTMF function:

Name	Description
OK	The function completed successfully. The default resolution for this output is SUCCESS.
Disconnected	On the established connection, the other party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

Detect DTMF

Detects a sequence of DTMF signals.

Detect DTMF: Parameters

The following table describes the Detect DTMF function parameters:


Name	Description
Delay Before Execution	<p>The time to wait before the function starts. It can be used for synchronization reasons (that is, to synchronize the Make Call function with the Receive Call one). It can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, Sleep 2000 in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) <p>The default is Static Expression, 20000 ms.</p>

DTMF Detection Settings	<ul style="list-style-type: none"> • Detect continuously for: Detects all the digits arrived within the specified period of time. The default value is 1000 seconds. • Detect exactly: Detects the specified number of digits. The default value is 6 DTMFs. • Use Talk Time (only for BHCA objective): Detects DTMS for the duration of the Talk Time parameter. • Detect DTMF Sequence: Detects the expected sequence (specified by selecting an entry in the DTMF Sequence Pool Items, or a specified Local Sequence).
Terminate Conditions	<ul style="list-style-type: none"> • Maximum delay between DTMFs: The maximum amount of time (in ms) allowed between consecutive digits for a proper detection. After this period elapses, the function terminates. The range of values is 0 to 999999 ms. The default value is 3000 ms. • First DTMF Timeout: The time (in ms) allowed for receiving the first digit. After this period elapses, the function exits on Timeout output. The range of values is 0 to 999999 ms. The default value is 2000 ms.

Detect DTMF: Advanced Detection Settings

The following table describes the Detect DTMF advanced detection settings:

Name	Description
Transmission Mode	<p>Detects DTMFs. The DTMFs can be detected In Band or Out of Band, using 2833 or the signaling layer:</p> <ul style="list-style-type: none"> • In Band – Using RTP media streaming. • Out of Band – Using 2833 EVENT Payload Format: The RTP Payload format used for carrying dual-tone multi frequency (DTMF) digits and other line and trunk signals as events. • Out of Band – Using 2833 TONE Payload Format: The RTP Payload format that can represent tones consisting of one or more frequencies. For details on RTP Payload, see RFC 2833 - RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals. • Use Global Settings – Use the settings from the Global Settings > Library Settings and Outputs > VoIP > RTP page.

 **Note:** In-band DTMF detection cannot be performed with audio codecs other than G.711 aLaw and G.711 uLaw.

Detect DTMF: Outputs

The following table describes the outputs available for the Generate DTMF function:

Output Name	Description
-------------	-------------

OK	The function completed successfully. The default resolution for this output is SUCCESS.
Timeout	This output is enabled if the request is not received within the specified timeout. The default resolution for this output is WARNING.
Disconnected	On the established connection, the other party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

Generate MF

Generates a specified MF sequence.

Generate MF: Parameters

The following table describes the Generate MF function parameters:

Name	Description
Delay Before Execution	<p>The time to wait before the function starts. It can be used for synchronization reasons (that is, to synchronize the Make Call function with the Receive Call one). It can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, Sleep 2000 in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) <p>The default is Static Expression, 20000 ms.</p>
MF Settings	<ul style="list-style-type: none"> • MF Duration: The time (in ms) required by a single tone to be generated. The range of values is 0 to 999999 ms. The default value is 200 ms. • Inter MF Interval: The maximum amount of time (in ms) between two consecutive generated MFs. The range of values is 0 to 999999 ms. The default value is 200 ms. • MF Amplitude: The attenuation (in dB) of the MF tone. The minimum attenuation is 0 dB (no attenuation) and the maximum is -40dB. The default value is -10 dB.
MF Sequence	Selects a specific MF sequence item from those defined in the DTMF Sequence Pool Items, or a Local Sequence. The default is 12345.

Generate MF: Advanced Playback Settings

These settings are identical to the Generate DTMF Advanced Playback Settings. For more information, see [Generate DTMF: Advanced Playback Settings](#).


Generate MF: Outputs

These outputs are identical to the Generate DTMF Outputs. For more information, see [Generate DTMF: Outputs](#).

Detect MF

Detects a sequence of MF signals.

All Detect MF function parameters are identical to these of the Detect DTMF function. For more information, see [Detect DTMF](#).

 **Note:** In-band MF detection cannot be performed with audio codecs other than G.711 aLaw and G.711 uLaw.

Generate Tone

Generates a custom tone.

Generate Tone: Parameters

The following table describes the Generate Tone function parameters:

Name	Description
Delay Before Execution	<p>The time to wait before the function starts. It can be used for synchronization reasons (that is, to synchronize the Make Call function with the Receive Call one). It can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, Sleep 2000 in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) <p>The default is Static Expression, 20000 ms.</p>
Tone Settings	<ul style="list-style-type: none"> • Tone Duration: The time (in ms) required to generate a single tone. The range of values is 0 to 999999 ms. The default value is 100 ms. • Tone Amplitude: The attenuation (in dB) of the DTMF tone. The minimum attenuation is 0 dB (no attenuation) and the maximum is -40dB. The default value is -10 dB.
Custom Tone	Tone Name: Selects a specific tone from the list.

Generate Tone: Advanced Playback Settings

All parameters in the Generate Tone Advanced Playback Settings tab are identical to the Generate DTMF Advanced Playback parameters. For more information, see [Generate DTMF: Advanced Playback Settings](#).

Generate Tone: Outputs

These outputs are identical to the Generate DTMF Outputs. For further information, see [Generate DTMF: Outputs](#).

Wait for Tone

Detects a custom tone.


Wait for Tone: Parameters

The following table describes the Wait for Tone function parameters:

Name	Description
Delay Before Execution	<p>The time to wait before the function starts. It can be used for synchronization reasons (that is, to synchronize the Make Call function with the Receive Call one). It can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, Sleep 2000 in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) <p>The default is Static Expression, 20000 ms.</p>
Tone Detection Settings	Select/Add the tone(s) to detect from those available in the Custom Tones Pool.
Terminate Conditions	Tone Timeout: The time (in ms) allowed for receiving a tone. After this period elapses, the function exits on Timeout output. The range of values is 0 to 999999 ms. The default value is 500 ms.

Wait for Tone: Advanced Detection Settings

All parameters in the Wait for Tone Advanced Detection Settings tab are identical to the Detect DTMF Advanced Detection parameters. For more information, see [Detect DTMF: Advanced Detection Settings](#).

 **Note:** In-band tone detection cannot be performed with audio codecs other than G.711 aLaw and G.711 uLaw.

Wait for Tone: Outputs

The following table describes the outputs available for the Wait For Tone function:



Output Name	Description
Timeout	The timeout value was exceeded or the received sequence does not correspond with the expected one. The default resolution for this output is WARNING.
Disconnected	On the established connection, the other party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

Talk

Plays the specified wave files across the established call.

Talk: Parameters

The following table describes the Talk function parameters:

Name	Description
Delay Before Execution	<p>The time to wait before the function starts. It can be used for synchronization reasons (that is, to synchronize the Make Call function with the Receive Call one). It can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, Sleep 2000 in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) <p>The default is Static Expression, 20000 ms.</p>
Overwrite activity settings	If selected, the selection of the audio clip and the playback settings specified at activity level are overridden by the current settings.
Clip	<p>An audio clip from the Resource Pool – either a simple .wav files or an speech clip – to be played back by the script function. Clicking  opens a window that permits you to select a .wav file.</p> <hr/> <p> Note: The eduration of speech clips, which are used for P.862 PESQ and P56 QoV computation, must not exceed 30 seconds.</p>
Output level	The clip output level (default -20 dbm).

Playback Settings	This area permits you to choose a clip playback duration.
Play for clip duration	If selected, the clip is played entirely for a specified number of times.
Repeat continuously for	If selected, the clip is played repeatedly for a specified duration of time.
Use Talk Time (for all objectives except channels)	If selected, the clip is played for the duration of the Talk Time parameter, an important parameter configured in the case of BHCA/CPS/LPS test objectives.
Use Global settings	If selected, the clip is played as specified in the RTP page of the Global Settings window.

Talk: Advanced Playback Settings

The following table describes the Talk function advance playback parameters:

Name	Description
Terminate conditions	Specifies a condition for stopping the wave playback as either of the following: <ul style="list-style-type: none"> DTMF: If selected, playing stops when the specified DTMF digit is received. The default DTMF is 0. Tone: If selected, playing stops when the specified custom tone is received. You can choose one of the available custom tones in the tones pool. MF: If selected, playing stops when the specified MF digit is received. The default value is 0.

Talk: Outputs

These outputs are identical to the Generate DTMF Outputs. For further information, see [Generate DTMF: Outputs](#).

Listen


Listens to an audio RTP stream for the specified duration.

 **Note:** Direct recording of wave files by the application is currently not supported.

Listen: Parameters

The following table describes the Listen function parameters:

Name	Description
------	-------------

Delay Before Execution	<p>The time to wait before the function starts. It can be used for synchronization reasons (that is, to synchronize the Make Call function with the Receive Call one). It can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, Sleep 2000 in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) <p>The default is Static Expression, 20000 ms.</p>
Overwrite activity settings	<p>If selected, the listen setting and QoV computation settings specified at activity level are overridden by the current settings.</p>
Listen Settings	<p>Specifies the listen duration as either of the following:</p> <ul style="list-style-type: none"> • Listen Duration: Specifies the period of time the listen operation is performed. The default value is 10000 ms. • Use Talk Time (only for BHCA objectives): Listens for an audio stream for the duration of the Talk Time parameter.
Perform QoV measurements	<p>If selected, performs P.862 PESQ and P56 QoV scores computation for the reference wave file specified in the Clip field.</p> <hr/> <p> Note: Speech clips duration must not exceed 30 seconds.</p>

Listen: Advanced Settings

The following table describes the Listen function advanced recording parameters:

Name	Description
Terminate conditions	<p>Specifies a condition for stopping the recording as either of the following:</p> <ul style="list-style-type: none"> • DTMF: If selected, playing stops when the specified DTMF digit is received. The default DTMF is 0. • Tone: If selected, playing stops when the specified custom tone is received. You can choose one of the available custom tones in the tones pool. • MF: If selected, playing stops when the specified MF digit is received. The default value is 0.

Listen: Outputs

These outputs are identical to the Generate DTMF Outputs. For further information, see [Generate DTMF: Outputs](#).

Voice Session

Plays and listens simultaneously (to) the specified wave files across the established call.

The audio files played by the Voice Session script function can be either simple wave files or audio clips (.wav) with a duration of maximum 30 seconds. Additionally, the files need to have the following characteristics:

- Coding: PCM, A-LAW, MU-LAW
- Sampling frequency: 8 kHz, monophonic, and having 8 bit/sample.

Voice Session: Talk Parameters

These parameters are identical to the Talk function parameters. For more information, see [Talk: Parameters](#).

Voice Session: Listen Parameters

These parameters are identical to the Listen function parameters. For more information, see [Listen: Parameters](#).

Voice Session: Advanced Playback Settings

The following table describes the Voice Session Advanced Playback parameters:

Name	Description
Terminate Conditions	This area specifies conditions for terminating the playback and the recording of audio clips.
Stop playback on first detected	If selected, this option enables you to specify a condition for stopping the audio playback as either of the following: <ul style="list-style-type: none"> • DTMF: If selected, playing stops when the specified DTMF digit is received. The default DTMF is 0. • Tone: If selected, playing stops when the specified custom tone is received. You can choose one of the available custom tones in the tones pool. • MF: If selected, playing stops when the specified MF digit is received. The default value is 0.
Stop recording on first detected	If selected, this option enables you to specify a condition for stopping the audio recording as either of the following: <ul style="list-style-type: none"> • DTMF: If selected, playing stops when the specified DTMF digit is received. The default DTMF is 0. • Tone: If selected, playing stops when the specified custom tone is received. You can choose one of the available custom tones in the tones pool. • MF: If selected, playing stops when the specified MF digit is received. The default value is 0.


Voice Session: Output Settings

The following table describes the outputs available for the Voice Session function:

Output Name	Description
OK	The function completed successfully. The default resolution for this output is SUCCESS.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

Multimedia Session



Plays audio and video files simultaneously or independently, depending on the current script function configuration, across an established SIP or H323 call.

 **Note:** The video files played by the Multimedia Session function need to have an MP4 format and be no larger than 80 MB.

Multimedia Session: Video Play Parameters

The following table describes the Multimedia Session Video Play parameters:

Name	Description
Delay Before Execution	<p>The time to wait before the function starts. It can be used for synchronization reasons (that is, to synchronize the Make Call function with the Receive Call one). It can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, Sleep 2000 in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) <p>The default is Static Expression, 20000 ms.</p>
Overwrite playback activity settings	For ease of configuration, playback settings for voice and video media can normally be configured at activity level. When this option is selected, the activity-level settings for the script function can be overridden.
Play audio	When the Overwrite playback activity settings option is selected, also selecting this option enables you to define an audio clip to play.

Play video	<p>When the Overwrite playback activity settings option is selected, also selecting this option enables you to define an mp4 video clip (AVC-format) to play.</p> <p> Note: In the case of VoIPSIP tests, you can also select SVC-format h264 video files to be played by this script function.</p>
Play for clip duration or Talk Time	<p>If selected, the clip(s) are played entirely, or for the duration of the Talk Time parameter (in case of a BHCA/CPS test objective).</p> <p> Note: In the case of BHCA/CPS objectives, the Talk Time and the number of channels are the two important parameters that must be specified for a given test objective value.</p>
Play for	If selected, plays the clip(s) for a specified duration of time.

Multimedia Session: Advanced Settings

The following table describes the Multimedia Session Advanced parameters:

Name	Description
Terminate conditions	<p>Stop playback on first detected: If enabled, playback stops if the specified DTMF, MF, or custom tone is detected (default Disabled).</p> <p>DTMF: If selected, listening stops when the specified DTMF digit is received. The default DTMF is 0.</p> <p>Tone: If selected, listening stops when the specified custom tone is received. You can choose one of the available custom tones in the tones pool.</p> <p>MF: If selected, listening stops when the specified MF digit is received. The default is 0.</p>

Multimedia Session: Output Settings

The following table describes the outputs available for the Multimedia Session function:

Output Name	Description
OK	The function completed successfully. The default resolution for this output is SUCCESS.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

MCPTT Send

Sends an MCPTT message.

Mission Critical Push To Talk (MCPTT) is an enhanced PTT (Push To Talk) service suitable for mission-critical scenarios. MCPTT is based on 3GPP Evolved Packet System (EPS) services and runs over LTE. IxLoad supports v13.5 of the MCPTT standard.

In IxLoad, MCPTT is implemented using scenario functions that you add to a VoIPSIP peer scenario. The scenario functions emulate an MCPTT client, and send and receive floor control messages (specifically Floor Control Protocol messages (MCPT) and Pre-Established Session Call Control Protocol messages (MCPC)).

All the MCPTT functionality is in the scenario functions; there are no MCPTT-specific settings at the VoIPSIP activity level.

MCPTT is supported on the S1/S11 and eNB stacks.

To use the MCPTT scenario functions in IxLoad, configure the VoIPSIP activity as follows:

1. Enable RTCP on the VoIPSIP activity.
2. Configure the SDP to negotiate MCPTT by adding the following to the Media line: `m=application [port] udp MCPTT`

MCPTT Send: General Parameters

The following table describes the MCPTT Send function parameters:

Name	Description																						
Message type	<p>Type of MCPTT message to send:</p> <table border="1"> <tr> <td data-bbox="354 317 695 380">Floor Request</td> <td data-bbox="699 317 1448 380">Request for permission to send media.</td> </tr> <tr> <td data-bbox="354 386 695 470">Floor Granted</td> <td data-bbox="699 386 1448 470">Informs the requesting floor participant that it has been granted the permission to send media.</td> </tr> <tr> <td data-bbox="354 476 695 560">Floor Deny</td> <td data-bbox="699 476 1448 560">Informs a floor participant that its floor request was rejected.</td> </tr> <tr> <td data-bbox="354 567 695 651">Floor Release</td> <td data-bbox="699 567 1448 651">Inform the floor control server that the floor can be released.</td> </tr> <tr> <td data-bbox="354 657 695 741">Floor Idle</td> <td data-bbox="699 657 1448 741">Informs a floor participant that no floor participant has permission to send media.</td> </tr> <tr> <td data-bbox="354 747 695 831">Floor Taken</td> <td data-bbox="699 747 1448 831">Informs the initiation of talk to the other group members.</td> </tr> <tr> <td data-bbox="354 837 695 963">Floor Revoke</td> <td data-bbox="699 837 1448 963">Informs the floor participant that has the permission to send media that its permission to send media is revoked.</td> </tr> <tr> <td data-bbox="354 970 695 1054">Floor Queue Pos Req</td> <td data-bbox="699 970 1448 1054">Request from a floor participant for information about its position in the floor request queue.</td> </tr> <tr> <td data-bbox="354 1060 695 1144">Floor Queue Pos Info</td> <td data-bbox="699 1060 1448 1144">Informs a floor participant of its position in the floor request queue.</td> </tr> <tr> <td data-bbox="354 1150 695 1339">Connect</td> <td data-bbox="699 1150 1448 1339"> <p>When sent from the originating side, confirms the establishment of an MCPTT call.</p> <p>When sent from the terminating side, initiates an MCPTT call.</p> </td> </tr> <tr> <td data-bbox="354 1346 695 1514">Disconnect</td> <td data-bbox="699 1346 1448 1514"> <p>When sent from the originating side, initiates ending of an MCPTT call.</p> <p>When sent from the terminating side, confirms ending an MCPTT call</p> </td> </tr> </table>	Floor Request	Request for permission to send media.	Floor Granted	Informs the requesting floor participant that it has been granted the permission to send media.	Floor Deny	Informs a floor participant that its floor request was rejected.	Floor Release	Inform the floor control server that the floor can be released.	Floor Idle	Informs a floor participant that no floor participant has permission to send media.	Floor Taken	Informs the initiation of talk to the other group members.	Floor Revoke	Informs the floor participant that has the permission to send media that its permission to send media is revoked.	Floor Queue Pos Req	Request from a floor participant for information about its position in the floor request queue.	Floor Queue Pos Info	Informs a floor participant of its position in the floor request queue.	Connect	<p>When sent from the originating side, confirms the establishment of an MCPTT call.</p> <p>When sent from the terminating side, initiates an MCPTT call.</p>	Disconnect	<p>When sent from the originating side, initiates ending of an MCPTT call.</p> <p>When sent from the terminating side, confirms ending an MCPTT call</p>
Floor Request	Request for permission to send media.																						
Floor Granted	Informs the requesting floor participant that it has been granted the permission to send media.																						
Floor Deny	Informs a floor participant that its floor request was rejected.																						
Floor Release	Inform the floor control server that the floor can be released.																						
Floor Idle	Informs a floor participant that no floor participant has permission to send media.																						
Floor Taken	Informs the initiation of talk to the other group members.																						
Floor Revoke	Informs the floor participant that has the permission to send media that its permission to send media is revoked.																						
Floor Queue Pos Req	Request from a floor participant for information about its position in the floor request queue.																						
Floor Queue Pos Info	Informs a floor participant of its position in the floor request queue.																						
Connect	<p>When sent from the originating side, confirms the establishment of an MCPTT call.</p> <p>When sent from the terminating side, initiates an MCPTT call.</p>																						
Disconnect	<p>When sent from the originating side, initiates ending of an MCPTT call.</p> <p>When sent from the terminating side, confirms ending an MCPTT call</p>																						
Delay before Execution	<p>Specifies a global function execution delay value as either:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) 																						

MCPTT Send: Output Settings

The following table describes the MCPTT Send Output Settings parameters:

Properties	Description
OK	The function completed successfully. The default resolution for this output is SUCCESS.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

MCPTT Wait

Receives MPCTT messages.

See [MCPTT Send](#) for a description of MCPTT support in IxLoad.

The following table describes the MCPTT Wait function parameters:

Name	Description
Message list	List of MCPTT messages that the function should expect to receive. If any one of the listed messages is received, the function exits successfully. To add a message to the list, click Add, then select the message to add. To remove a message from the list, select the message, then click Remove.
Timeout	The time (in ms) the function waits for the received message to match the template. If this time period terminates without having a match, the function enables the Timeout output. It can be specified as follows: <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms)


MCPTT Wait: Output Settings

The following table describes the MCPTT Wait Output Settings parameters:

Properties	Description
<message>	MCPTT message received successfully.
Timeout	A timeout occurred while the function was executing. The default resolution for this output is WARNING.
Error	The function execution has returned an error. The default resolution for this output is FAILED.


T.38 Fax Session

Sends or receives the specified fax data file across an established SIP call.

 **Note:** The T.38 Fax Session script function can only be used for calls negotiated using the SIP signaling protocol.

T.38 Fax Session: General Parameters

The following table describes the T.38 Fax Session function parameters:

Name	Description
Send fax	If selected, the image specified in the adjoining field is sent over a simple fax session. To specify an image, click  and choose an image file from the Resource Pool.
Receive fax	If selected, the script function executes a fax reception.
Overwrite activity settings	If selected, the activity-level fax parameters (specified in the T.38 and T.30 tabs) are overridden by function level settings. You can define an image to send/receive other than that configured at activity level in the T.30 tab and override the following send/receive parameters:
Send Parameters	
Coding	The highest coding scheme available for compressing the page data when sending. Possible values are MH, MR, and MMR (default).
Data rate (kbps)	The data rate for sending, which is any of the following: V.27 ter 2.4 Kbps, V.27 ter 4.8 Kbps, V.17 7.2 Kbps, V.17 9.6 Kbps, V.17 12 Kbps, V.17 14.4 Kbps, V.29 7.2 Kbps, V.29 9.6 Kbps, V.34 16.8 Kbps, V.34 19.2 Kbps, V.34 21.6 Kbps, V.34 24 Kbps, V.34 26.4 Kbps, V.34 28.8 Kbps, V.34 31.2 Kbps and V.34 33.6 Kbps available data rates.
Page size	The sent page size, any of the following: A4 (210x297 mm), B4 (255x364 mm), and A3 (297x420 mm) available formats.
MSLT	The minimum transmission time of one coded scan line (values (lower value than specified are accepted). This value is determined automatically, based on DIS.
Protocol	The protocol used for fax sending. The available protocols are non-ECM and ECM (default).
Resolution	The horizontal and vertical resolution of the page image. Possible values is any combination of the following: <ul style="list-style-type: none"> - R8x3.85 lines/mm (default) - R8x7.7 lines/mm - R8x15.4 lines/mm - 200x200 dots/inch

Send CNG	Specifies if the CNG message is sent or not (default yes).
Receive Parameters	
Coding	The highest coding scheme available for compressing the page data when receiving. Possible values are MH, MR, and MMR (default).
Page size (up to)	The maximum size of the received page. Possible values are A4 (210x297 mm) (default), B4 (255x364 mm), and A3 (297x420 mm).
MSLT	The minimum transmission time of one coded scan line (values (lower than specified are accepted). The possible values are: - 0 ms T7.7 = T3.85 (default)
Protocol	The protocol used to receive fax. The available protocols are non-ECM and ECM (default).
Send CED before DIS	If selected, this enables the answering fax to send a CED (CallEd station Identification) signal. This tone is generated to allow a human participant to realize that a machine is present on the other end of the call (default Disabled).
Modulation	Specify the protocols available for receiving. Possible values are as follows: - V.27 - V.27/V.29 - V.27/V.29/V.17 - V.27/V.29/V.17/V.34 (default)
Receive Resolution	This area specifies a list of resolutions (horizontal and vertical) to choose from. Possible values are any combination of the following: - R8x3.85 lines/mm - R8x7.7 lines/mm - R8x15.4 lines/mm - 200x200 dots/inch

T.38 Fax Session: Output Settings

The following table describes the T.38 Fax Session Output Settings parameters:

Properties	Description
OK	The fax session was completed successfully. The default resolution for this output is SUCCESS.
Error	The function execution has returned an error. The default resolution for this output is FAILED.


Path Confirmation

After a voice path has been established using VoIP signaling, this script function confirms the existence of a viable voice path between two VoIP endpoints by simultaneously generating and detecting DTMF, MF or tone sequences. Path confirmation is performed in-band (using RTP audio streaming) or out-of-band (using RFC 2833 events).

Path confirmation sequences can be generated once or can be looped through for a specified period of time. The existence of the voice path is validated if all the sequences can be detected by the endpoints in the specified timeout intervals.

The Path Confirmation script function is full duplex, meaning that the generation and detection of sequences can be performed at the same time.



- The Path Confirmation function does not support cadenced tones.
- In-band path confirmation cannot be performed with audio codecs others than G.711 aLaw and G.711 uLaw.


 **Note:** After the Path Confirmation function starts execution and before tones generation / detection is actually performed, a synchronization procedure for functions on different scenario channels is launched, having a maximum duration of 2000 ms.

Path Confirmation: Parameters

The following table describes the Path Confirmation function parameters:


Name	Description
Delay before execution	<p>The time to wait before the function starts. This delay can be used for synchronization purposes, such as, for example, to synchronize the call originating side with the call terminating side. It can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression: This fixed delay before execution can be expressed as a value in milliseconds (ms), or as a formula. • Random Between Expressions: This is a random delay in the specified interval that is used to simulate real-life conditions. It is expressed in milliseconds (ms). • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 1000, 2000, in milliseconds (ms) <p>The default is Static Expression, 20000 ms.</p>

Path Confirmation Method	<p>Specifies the path confirmation method as using either DTMFs, MFs, or custom tones sequences.</p> <p>The path confirmation initiator sends the specified digits sequence and expects to receive the same sequence, while the path confirmation receiver expects the specified digits and then sends them back.</p> <p>When you choose to send sequences as DTMFs or MFs, you need to click the corresponding  to define the sequence as a VoIP Plug-in Resource Pool predefined sequence or by entering the digits directly into a field (local sequence).</p> <p>When you choose to perform path confirmation using custom tones, you need to look up the predefined editable tones from the Resource Pool > Custom Tones category – such as, for example, DIGIT_1, DIGIT_2, DIGIT_3, DIGIT_4, DIGIT_5, DIGIT_6, DIGIT_7, DIGIT_8, DIGIT_9, or DIGIT_0 – and define the sequence by specifying only the corresponding digits. For example, a valid sequence comprising five tones is specified using the 01234 string.</p> <hr/> <p> Note: DTMF sequences can be made up of 0-9, A, B, C, D digits, while MF sequences support only 0-9, A, B, C digits.</p>
--------------------------	---

 **Note:** When specifying a DTMF, MF, or tone sequence using expressions, you must ensure that every channels pair evaluates the expressions to the same value, otherwise the path confirmation sequence fails.

Path Confirmation: Tone Detection/Generation Page


The following table describes the tone detection/generation parameters:

Name	Description
Use values from Global Settings	If selected, the values of all other parameters on this page are taken from the Global Settings > Library Settings and Outputs > VoIP > RTP page.
DTMF/MF/Tone Generation	<p>DTMF/MF/Tone generation settings, as follows:</p> <ul style="list-style-type: none"> • Tone Duration: The duration (in ms) of a single tone (DTMF/MF/custom tone). The range of values is 40 to 59960 ms. The default value is 200 ms. • Inter Tone Interval: The maximum amount of time (in ms) between two consecutive generated DTMFs/MFs/Custom Tones. The range of values is 40 to 59960 ms. The default value is 200 ms. <hr/> <p> Note: The sum of the tone duration and the inter tone interval is required to be less than 60000 ms.</p> <hr/> <ul style="list-style-type: none"> • Tone Amplitude: The attenuation (in dB) of the DTMF/MF/custom tone. The minimum attenuation is 0 dB (no attenuation) and the maximum is -40 dB. The default value is -10 dB.

DTMF/MF/Tone Detection	<p>DTMF/MF/Tone detection settings, as follows:</p> <ul style="list-style-type: none"> • First Sequence Timeout: The time (in ms) allowed for detecting the first DTMF/MF/custom tones sequence. After this period elapses, the function exits on Timeout output. The range of values is 200 to 999999 ms. The default value is 5000 ms. • Intersequence timeout: The maximum amount of time (in ms) allowed between consecutive DTMFs/MFs/custom tones sequences for a proper detection. After this period elapses, the function exits on Timeout output. The range of values is 200 to 999999 ms. The default value is 5000 ms.
------------------------	---

Path Confirmation: Advanced Settings

The following table describes the Path Confirmation function advanced settings:

Name	Description
Transmission Mode	<p>Specifies the DTMF/MF/TONE transmission mode as one of the following:</p> <ul style="list-style-type: none"> • In Band – Using RTP media streaming. • Out of Band – Using 2833 EVENT Payload Format: The RTP Payload format used for carrying dual-tone multi frequency (DTMF) digits and other line and trunk signals as events. • Out of Band – Using 2833 TONE Payload Format: The RTP Payload format that can represent tones consisting of one or more frequencies. For details on RTP Payload, see RFC 2833 - RTP Payload for DTMF Digits, Telephony Tones and Telephony Signals. • Use Global Settings – Use the settings from the Global Settings > Library Settings and Outputs > VoIP > RTP page. <hr/> <p> Note: A G.711 aLaw or uLaw codec must be used for both in-band and out-of-band path confirmation. If a G.711 codec cannot be found, the path confirmation operation cannot start. A 2833 Event/Tone codec must be used for out of band Event/Tone media transmission.</p>
Playback	<p>Specifies how many times the path confirmation sequence is executed, as either of the following options:</p> <ul style="list-style-type: none"> • Execute once: The sequence is played once. • Execute for: The sequence is executed for a user defined period of time, expressed in seconds, minutes or hours. The default value is 10 seconds. • An expression can be also entered in this edit box. At run time, the expression is evaluated and its value is considered in milliseconds. • Use Global Settings: Uses the path confirmation sequence settings from the Global Settings > Library Settings and Outputs > VoIP > RTP page.

Path Confirmation: Output Settings

The following table describes the Path Confirmation output parameters:


Output Name	Description
OK	The function completed successfully. The default resolution for this output is SUCCESS.
Timeout	This output is enabled if the request is not received within the specified timeout. The default resolution for this output is WARNING.
Disconnected	On the established connection, the other party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.
Error	The function has returned an internal error. The default resolution for this output is FAILED.
Mismatch	The function has detected a digit that is not in the expected path confirmation sequence.


RTP Control

IxLoad RTP script functions support Non-Blocking Execution, meaning that, after a RTP script function has been initialized and started, the execution flow advances to the next function in scenario, without waiting for the current command to finish. When running in Non-Blocking Execution mode, RTP script functions perform their tasks in background, allowing simultaneous handling of other signaling and media functions.

The RTP Control function has the purpose of controlling one (the previous) RTP script function when executing a RTP script function that has non-blocking behavior enabled.

If non-blocking execution is not enabled and the RTP Control script function is used in a test scenario, it exits on a SUCCESS output.

 **Note:** An RTP script function running in non-blocking mode always exits on the OK/SUCCESS output, while the 'true' result of the function execution is obtained by evaluating the output of the RTP Control function. A special case is the WaitTone script function which does not have an OK output. A non-blocking WaitTone function always exits on the output of the expected first tone.

 **Note:** Only one RTP script function can be active at any one time during test execution. If the scenario flow contains an RTP script function with non-blocking behavior and this function is followed by a second RTP function, when the execution flow reaches the second function, the first function is terminated, such that no RTP functions execute in parallel.

RTP Control: Parameters

The following table describes the RTP Control function parameters:


Name	Description
------	-------------

Delay before execution	<p>The time to wait before the function starts. This delay can be used for synchronization purposes, such as, for example, to synchronize the call originating side with the call terminating side. It can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression: This fixed delay before execution can be expressed as a value in milliseconds (ms), or as a formula. • Random Between Expressions: This is a random delay in the specified interval that is used to simulate real-life conditions. It is expressed in milliseconds (ms). • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 1000, 2000, in milliseconds (ms) <p>The default is Static Expression, 20000 ms.</p>
Control Actions	<ul style="list-style-type: none"> • Check for RTP Completion: Checks the status of an RTP function executing in non-blocking mode, and returns the status code (see RTP Control: Output Settings) accordingly. • Terminate RTP: Forcefully terminates a running non-blocking RTP function.

RTP Control: Output Settings

The following table describes the RTP Control output parameters:

Output Name	Description
RTP Running	This output is selected when a non-blocking RTP script function is running in background (cannot be returned if the Terminate RTP option was configured for the RTP Control function). The default resolution for this output is SUCCESS.
RTP Not Running	The previous RTP function is no longer running in the background and terminated successfully. No RTP function was started before. The default resolution for this output is SUCCESS.
Error	The RTP function running in the background exited with an error. The default resolution for this output is FAILED.
Timeout	The RTP function running in the background has finished with a timeout condition. The default resolution for this output is WARNING.
Disconnected	The RTP function running in the background has finished with a disconnect condition. The default resolution for this output is WARNING.

 **Note:** The Wait Tone script function is an exception and in case the output for a Wait Tone is a detected tone (no timeout or error), a variable called RTP_NB_DETECTED_TONE is filled with the name of the detected tone and RTP Control function returns RTP Not Running.

Forceful RTP Function Termination and Statistics

An RTP function that is running in the background is forcefully terminated in any of the cases below:

- When SDP renegotiation changes the RTP parameters (codec, etc.): In this case, the RTP Skipped Functions statistic is incremented.
- When the RTP Control function terminates it: In this case, the RTP Skipped Functions statistic is incremented.
- When a subsequent RTP script function terminates it: In this case, the RTP Skipped Functions statistic is incremented.
- When an EndCall function is executed while the RTP function is running and the EndCall procedure must be initiated: In this case, the RTP Disconnected Functions statistic is incremented.
- When Graceful ramp-down is started: In this case, an RTP function is stopped/skipped and the RTP Skipped Functions statistic is incremented.
- When normal ramp-down is started: In this case, no statistics are incremented.
- When the call ends and the EndCall procedure is initiated: In this case, the RTP Errors and the Failed Playbacks/Failed Records statistics are incremented.

Flow Statistics

Flow statistics are updated normally, since a non-blocking RTP function always exits on the SUCCESS output and the real result of the function execution is obtained from the RTP Control function.

SDP Renegotiation

If SDP re-negotiation is done after an RTP non-blocking function has started, the function is stopped and RTP context is re-opened with new parameters. The stopped RTP function is not restarted after the RTP context is re-opened.

If after a SDP renegotiation nothing has changed, then the existing RTP context is unchanged, and any running RTP non-blocking function is not stopped.

Warnings and Errors

- If non-blocking execution is enabled, but the test does not contain a RTP Control script function, the following warnings are shown when applying the test configuration:
 - RTP Non-blocking execution is enabled, but the test does not contain a RTP Control script function. The status of the RTP script functions execution will not be available.
- If non-blocking execution is disabled but the scenario does contain a RTP Control script function the following warning are shown when applying the test configuration:
 - RTP Non-blocking execution is disabled, but the test contains a RTP Control script function. RTP script functions execution always exit on RTP Not Running output.

VoIP Flow Functions Library

The VoIP Flow Test Library includes the following script functions:

- [Start](#)
- [Stop](#)
- [Variable Set](#)
- [Variable Test](#)
- [Sleep](#)
- [Procedure](#)
- [Exit Procedure](#)
- [Counter Op](#)
- [Test Time](#)
- [Log Message](#)
- [Dump Variables](#)
- [Error Handler](#)

Start

Description

Indicates the beginning of a test scenario flow on the channel.



Note: Only one Start function is allowed per channel.

Parameters

No parameters

Stop

Description

Indicates the end of an execution thread or the end of the entire test scenario.

Parameters

No parameters


Variable Set

Description

Declares and sets the variable values to use in the test scenario.

For each variable, you must specify a name, a type (long, string, array of long, or array of string) and a value, or an expression composed of numbers and variables.


When specifying an expression instead of a value, at execution time, the expression is evaluated to a value that is assigned to the variable.

 **Note:** The Variable Set function is generally used conjointly with the Variable Test function that enables you to evaluate custom expressions.

Parameters

The user-defined variables used in this function block, of whom only the selected (tagged) variables are considered. The tagged variables display first.

Variables can be assigned numeral values, or expressions comprising numerals, operators, functions and/or variables.

 **Note:** Evaluating an expression assigned to a variable is done by the Expression Evaluator, whose supported operators and operations are described in Appendix B.

A couple of expression examples are given in the following table:

Expression	Comments
\$MapPos + 2	Used in a Variable Set function to increment the value of the \$MapPos variable (\$MapPos is equal to the current mapping position number).
\$NewMediaPort = \$VoipMediaBaseAddress + \$UnitCh	When used in a custom SDP message, this expression assigns to the \$NewMediaPort variable the value of the base RTP port (usually 10000) increased by value of the channel index. For example, unit channel 0 will have a media port equal value of 10000, while unit channel 10 will have the RTP port value of 10010.

Variable Test

Description

This function assesses a series of expression sequences, each of whom adds a new output to the script function. At the time of the script function execution, the expression sequences, composed of numerals, operators, functions, and/or variables, are evaluated to either True or False.

The evaluation starts with the first expression sequence and continues until a sequence is found that evaluates to True. When such an expression sequence is found, the execution flow goes on with the block connected to the output related to that sequence.




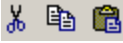


If none of the assessed sequences return true, the execution flow goes on with the block connected to the Mismatch output.


Parameters

The associated parameters are the series of expressions evaluated in the function at test execution time.

Each expression has a name that generates an associated output, and an expression sequence. While evaluating the expressions in turn, if the script function encounters an expression that evaluates to True, the Variable Test function exits on the corresponding output.

Operations available for creating and managing expressions are given in the following table.

Operation	Description
	Opens a window that permits you to define an expression name (corresponding to an output) and an expression sequence to be evaluated at test execution time.
	Edits an existing expression by opening the same window as that displayed by clicking the Add new expression  .
	Performs common cut, copy paste operations.
	Deletes an expression entry.
	Because the expression sequences order is relevant (higher ranked expressions are evaluated first), ordering the list is possible using these commands.

 **Note:** Evaluating an expression sequence is done by the Expression Evaluator, whose supported operations and operators are described in Appendix B.

Sleep

Description

This script function inserts a static or random delay to the execution flow.

Parameters

The associated parameter is the delay applied to the execution flow. The delay may be:

- Static – the fixed delay period in milliseconds (ms). It takes values ranging from 0 to 6000000 ms. The default value is 500 ms.
- Random – random delay values within a chosen range (between start and stop), in ms. Both start and stop durations take values ranging from 0 to 6000000 ms. The default values are random between start = 500 ms and stop = 5000 ms.

Procedure

Description

Declares a procedure in the current test scenario. A procedure is used to encapsulate several functions in a single script for further reuse.

Parameters

This function block has no parameters.

Exit Procedure

Description

Determines the output of a procedure. A procedure may have one or more outputs. Each Exit Procedure function represents a separate output for the Procedure function block.

Parameters

The optional output name. It is recommended to use this option and to give suggestive names to each output. Thus, you can make the difference between several outputs in the same procedure. The name entered here is the output name in the Procedure function block.

Counter Op

Description

Inserts user-defined statistic counters, which can be incremented, decremented, or reset. The counter value can be subject to a post-execution analysis process.

Parameters

The user-defined statistic counters.

The list of user-defined statistic counters can be modified and different operations can be applied to them:

- Increment
- Decrement
- Increment with expression
- Decrement with expression
- Set value (expression)
- Reset


Test Time

Description

Assesses the current time. It can be used to control the execution flow depending on time; the user can add more time intervals, each representing a new output for the function. The output is selected by comparing each time interval with the application's local time. The Mismatch output is selected if no output includes the local time (and day, if used).

For a description of the Test Time function parameters, see the following table:

Name	Description
------	-------------

Time Interval (list)	The time interval to be compared with the local time. Its duration can span 1 second (s) to a whole week. There are two cases: <ul style="list-style-type: none"> • Time interval in the same day, specified as two time limits, in hh:mm:ss format (for example, 15:23:32-23:34:55). • Time interval in different days of the week, specified as two time limits in day_of_the_week hh:mm:ss format (for example, Sun 15:23:32-Mon 23:34:55).
Use Day Of The Week	If selected, the user can compare the time and days of the week. The default is Disabled .
Between ... And ...	The two-time stamps (and days of the week, if used) defining the time interval. You can specify the time in the hh:mm:ss AM/PM format. <hr/>  Note: The time displays in the Time Interval list in 24h format.

Log Message

Description

Enables you to add messages that are included in the execution log. These messages may include variables that are substituted at runtime with the appropriate values.

Note, however, that expressions including messages will not be fully evaluated, only value substitution will be done. For example, a log message such as "This is \$Iter+1" will not evaluate the \$Iter+1 expression, but only substitute the value of the \$Iter variable.

Parameters

The message to include in the log.

Dump Variables

Description

Generates a log containing all variables from all engines and the user-defined variables. For engines not available for the channel containing this script function, the variables have empty values.

Parameters

N/A

Error Handler

Description

This script function can be used to minimize the number of connectors in a script. If you include this function in a scenario and the other functions do not have the 'Error,' 'Timeout,' 'Disconnected,' or 'Invalid' outputs connected, the Error Handler function is automatically called.

Parameters

By double-clicking the function in the scenario editor, the Error Handler Properties display, enabling you to access the parameters described in the following table:

Parameter	Description	Default Value
Properties	Select the outputs to display (at least one) – choose from the following list of checkboxes: <ul style="list-style-type: none"> • Error • Timeout • Disconnected • Invalid • Transport Failure • Found • Not Found 	Enabled Error and Timeout check boxes
Output Settings	Error – the function returns an internal error.	Failed
	Time-out – no function is running.	Warning

 **Note:** Make sure that there is a single Error Handler function per channel.

VoIP H323 RAS Library

The VoIP H.323 RAS Library includes the following script functions:

- [H323 Register](#)
- [H323 UnRegister](#)

H323 Register

Registers an H323 endpoint with a gatekeeper.

Register Properties: Parameters

The following table describes the Register function parameters:

Name	Description
------	-------------

Delay	<p>Delays the function execution by a value specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 2000, in milliseconds (ms)
Time to Live	<p>The time after which registration expires (milliseconds), specified as either of the following:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Delay Between Digits, in milliseconds (ms) • PHONE_WAIT_TIME, in milliseconds (ms) • MGCP Timeout, in milliseconds (ms)

Register Properties: Output Settings

The following table describes the outputs available for the Register function:

Name	Description
OK	The function completed successfully. The default resolution for this output is SUCCESS.
Rejected	Rejects the endpoint request. The default resolution for this output is SUCCESS.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

H323 UnRegister

Unregisters a currently registered H323 endpoint.

H.323 Unregister: Parameters

The following table describes the Unregister function parameters:

Name	Description
------	-------------

Delay	<p>The time to wait before the function starts. It can be used for synchronization reasons (that is, to synchronize the Make Call function with the Receive Call one). It can be specified as follows:</p> <ul style="list-style-type: none">• Static Expression, in milliseconds (ms)• Random Between Expressions, in milliseconds (ms)• Sleep 1000, in milliseconds (ms)• GetCallInfo Delay, in milliseconds (ms)• Detect DTMF delay, in milliseconds (ms)• Generate DTMF delay, in milliseconds (ms)• Sleep 2000, in milliseconds (ms) <p>The default is Static Expression, 0 ms.</p>
-------	--

H323 Unregister: Output Settings

The following table describes the outputs available for the Unregister function:

Name	Description
OK	The function completed successfully. The default resolution for this output is SUCCESS.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

VoIP H248 Functions Library

The VoIP Megaco/H.248 Test Library includes the following script functions that emulate complete transactions, comprising both command requests and replies that manipulate contexts, terminations, events, and signals.

H.248 MGC Library


- [Add](#)
- [Modify](#)
- [Move](#)
- [Subtract](#)
- [AuditVal](#)
- [AuditCap](#)
- [SrvChange \(MGC\)](#)
- [Wait Notify](#)
- [Wait SrvChange \(MGC\)](#)
- [Wait Requests \(MGC\)](#)





Add

This function sends a transaction request with one or more Add commands and waits for the transaction reply. If the corresponding H248 activity is configured with the **Use TransactionResponseAck** option, the script function also sends a final acknowledgment.

Add: Tx Request

The following table describes the request parameters of the Add script function:

Name	Description
Context ID	<p>The request context ID, which can be any of the following:</p> <ul style="list-style-type: none"> • - : This denominates the NULL context. • \$: This denominates an arbitrary context that is chosen by the device. • * : This denominates all contexts. • <AUTO> : This denominates the context associated with the current user. <hr/> <p> Note: Numerical values and expressions (enclosed in []) are allowed.</p>
Send ContextRequest	<p>If enabled, a user-configurable context request is sent. To edit the request, select this option and click Edit.</p>


<p>Send ContextAttribAuditRequest</p>	<p>If enabled, a user-configurable context attributes audit request is sent. To edit the request, select this option and click Edit.</p> <p>For each command to be sent, a property page named Add Request#n appears and permits you to configure the following command parameters:</p> <p>The request termination ID, which can be any of the following:</p> <ul style="list-style-type: none"> • ROOT : This denominates the ROOT termination. <ul style="list-style-type: none"> ■ \$: This denominates an arbitrary termination that is chosen by the device. ■ * : This denominates all terminations. ■ <AUTO-PHYSICAL> : This denominates the physical termination of the context associated with the current user. ■ <AUTO-RTP1> : This denominates the first RTP termination of the context associated with the current user. ■ <AUTO-RTP2> : This denominates the second RTP termination of the context associated with the current user. • The following options related to flags: <ul style="list-style-type: none"> ■ Optional: If selected, the command’s optional flag is set. ■ Wildcard Return: If selected, the command’s wildcard flag is set. • A command-specific list of descriptors (only the descriptors that can be contained in the current command) that are sent in the current command request. When a descriptor is selected, the pane below displays an editable tree representation of the descriptor. See Appendix C for more information on editing message descriptors.
	<p>Click Add to add a new Add command and a corresponding property page as a separate tab (named Add Request#n).</p>
	<p>Click Delete to delete the current Add command property page. You cannot delete if there is only one property page.</p>
	<p>Click Preview to view a window that shows the current transaction request code. The following options are available:</p> <ul style="list-style-type: none"> • Encode Type: Specifies the display mode as Normal, Compact, or Pretty. • Version: Displays the selected protocol request version.
	<p>Click Templates to open a window that allows the import of a transaction from a list of existing templates. After a template is selected, you can view the contents of the template, the folder location, and a short description of the template.</p>



Click **Import** to open a window that allows the import of a H248 message in plain text format.

Add: Rx Reply

The following table describes the expected reply parameters of the Add script function:

Name	Description
Context ID	<p>The context ID of the expected reply, which can be any of the following:</p> <ul style="list-style-type: none"> • - : This denominates the NULL context. • \$: This denominates an arbitrary context that is chosen by the device. • * : This denominates all contexts. • <AUTO> : This denominates the context associated with the current user. • <ANY> : If selected, the context ID value is ignored. <hr/> <p> Note: Numerical values and expressions (enclosed in []) are allowed.</p>
Expect message error	If selected, the expected reply message contains an error descriptor at message level. The expected error code may be specified in the adjacent field.
Expect transaction error	If selected, the expected reply message contains an error descriptor at transaction level. The expected error code may be specified in the adjacent field.
Expect action error	If selected, the expected reply message contains an error descriptor at action level. The expected error code may be specified in the adjacent edit field.
Expect transaction/action reply	If selected, the expected reply message can be edited in the pane below.

<p>Expect ContextReply</p>	<p>If enabled, context parameters following a context request are expected. The expected context parameters can be edited by clicking Edit.</p> <p>For each expected command reply, a property page (Add Reply#n) is displayed and permits you to edit the following reply parameters:</p> <ul style="list-style-type: none"> • The termination ID of the reply, which can be any of the following: <ul style="list-style-type: none"> ■ ROOT : This denominates the ROOT termination. ■ \$: This denominates an arbitrary termination that is chosen by the device. ■ * : This denominates all terminations. ■ <AUTO-PHYSICAL> : This denominates the physical termination of the context associated with the current user. ■ <AUTO-RTP1> : This denominates the first RTP termination of the context associated with the current user. ■ <AUTO-RTP2> : This denominates the second RTP termination of the context associated with the current user. ■ <ANY> : If selected, the termination ID value is ignored. • A command-specific list of descriptors (only the descriptors that can be contained in the current command) that are expected in the current command reply. When a descriptor is selected, the pane below displays an editable tree representation of the descriptor. See Appendix C for more information on editing descriptors.
----------------------------	--

Add: Parameters

The following table describes the execution parameters of the Add script function:

Name	Description
<p>Delay Before Execution</p>	<p>Delays the function execution by a duration that can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, Sleep 2000 in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms)

Timeout	<p>The time, in milliseconds (ms), that the script function waits for it to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled.</p> <p>It can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Delay Between Digits, in milliseconds (ms) • PHONE_WAIT_TIME, in milliseconds (ms) • MGCP Timeout, in milliseconds (ms)
---------	--

Add: Output Settings

The following table describes the outputs available for the Add script function:

Output Name	Description
OK	The function completed successfully. The default resolution for this output is SUCCESS.
Timeout	The default resolution for this output is WARNING.
Error	The function has returned an internal error. The default resolution for this output is FAILED.

Modify

This function sends a transaction request with one or more Modify commands and waits for the transaction reply for the specified timeout. If the corresponding H248 activity is configured with the **Use TransactionResponseAck** option, the script function also sends a final acknowledgment. For description on the configuration parameters, see [Add: Tx Request](#), [Add: Rx Reply](#), [Add: Parameters](#), and [Add: Output Settings](#).

Move

Sends a transaction request with one or more Move commands and waits for the configured with the **Use TransactionResponseAck** option, the script function also sends a final acknowledgment. For description on the configuration parameters, see [Add: Tx Request](#), [Add: Rx Reply](#), [Add: Parameters](#), and [Add: Output Settings](#).

Subtract

Sends a transaction request with one or more Subtract commands and waits for the transaction reply for the specified timeout. If the corresponding H248 activity is configured with the **Use TransactionResponseAck** option, the script function also sends a final acknowledgment. For description on the configuration parameters, see [Add: Tx Request](#), [Add: Rx Reply](#), [Add: Parameters](#), and [Add: Output Settings](#).

AuditVal

Sends a transaction request with one or more AuditValue commands and waits for the transaction reply for the specified timeout. If the corresponding H248 activity is configured with the **Use TransactionResponseAck** option, the script function also sends a final acknowledgment. For description on the configuration parameters, see [Add: Tx Request](#), [Add: Rx Reply](#), [Add: Parameters](#), and [Add: Output Settings](#).

AuditCap

Sends a transaction request with one or more AuditCapabilities commands and waits for the transaction reply for the specified timeout. If the corresponding H248 activity is configured with the **Use TransactionResponseAck** option, the script function also sends a final acknowledgment. For description on the configuration parameters, see [Add: Tx Request](#), [Add: Rx Reply](#), [Add: Parameters](#), and [Add: Output Settings](#).

SrvChange (MGC)


Sends a transaction request with one or more ServiceChange commands and waits the transaction reply for the specified timeout. If the corresponding H248 activity is configured with the **Use TransactionResponseAck** option, the script function also sends a final acknowledgment. For description on the configuration parameters, see [Add: Tx Request](#), [Add: Rx Reply](#), [Add: Parameters](#), and [Add: Output Settings](#).




Wait Notify



Waits for a transaction request with one or more Notify commands for the specified time (timeout value) and then sends a configured transaction reply. If the corresponding H248 activity is configured with the **Use TransactionResponseAck** option, the script function also waits for a final acknowledgment.

Wait Notify: Rx Request

The following table describes the request parameters of the Wait Notify script function:


Name	Description
Context ID	<p>The request context ID, which can be any of the following:</p> <ul style="list-style-type: none"> • - : This denominates the NULL context. • \$: This denominates an arbitrary context that is chosen by the device. • * : This denominates all contexts. • <AUTO> : This denominates the context associated with the current user. <hr/> <p> Note: Numerical values and expressions (enclosed in []) are allowed.</p>

Expect ContextRequest	If enabled, a user-configurable context request is expected. To edit the request, select this option and click Edit .
Expect ContextAtribAuditRequest	<p>If enabled, a user-configurable context attributes audit request is expected. To edit the request, select this option and click Edit.</p> <p>For each expected command request, a property page named Add Request#n appears and permits you to configure the following command parameters:</p> <p>The command termination ID, which can be any of the following:</p> <ul style="list-style-type: none"> • ROOT : This denominates the ROOT termination. <ul style="list-style-type: none"> ▪ \$: This denominates a termination that is chosen by the device. ▪ * : This denominates all terminations. ▪ <AUTO-PHYSICAL> : This denominates the current physical termination of the context associated with the current user. ▪ <AUTO-RTP1> : This denominates the first RTP termination of the context associated with the current user. ▪ <AUTO-RTP2> : This denominates the second RTP termination of the context associated with the current user. ▪ <ANY> : When selected, the termination ID value is ignored. • The following command options: <ul style="list-style-type: none"> ▪ Optional: If selected, the command's optional flag is set. ▪ Wildcard Return: If selected, the command's wildcard flag is set. • A command-specific list of descriptors (only the descriptors that can be contained in the current command) that are sent in the current command request. When a descriptor is selected, the pane below displays an editable tree representation of the descriptor. See Appendix C for more information on editing message descriptors.
	Click Add to add a new Add command and a corresponding property page as a separate tab (named Add Request#n).
	Click Delete to delete the current Add command property page. You cannot delete if there is only one property page.
	Click Preview to view a window that shows the current transaction request code. The following options are available: <ul style="list-style-type: none"> • Encode Type: Specifies the display mode as Normal, Compact, or Pretty. • Version: Displays the selected protocol request version.

	Click Templates to open a window that allows the import of a transaction from a list of existing templates. After a template is selected, you can view the contents of the template, the folder location, and a short description of the template.
	Click Import to open a window that allows the import of a H248 message in plain text format.

Wait Notify: Tx Reply

The following table describes the reply parameters of the Wait Notify script function:

Name	Description
Context ID	The reply context ID, which can be any of the following: <ul style="list-style-type: none"> • - : This denominates the NULL context. • \$: This denominates an arbitrary context that is chosen by the device. • * : This denominates all contexts. • <AUTO> : This denominates the context associated with the current user. <hr/>  Note: Numerical values and expressions (enclosed in []) are allowed.
Send message error	If selected, the sent reply message contains an error descriptor at message level. The error code may be specified in the adjacent field.
Send transaction error	If selected, the sent reply message contains an error descriptor at transaction level. The sent error code may be specified in the adjacent field.
Send action error	If selected, the sent reply message contains an error descriptor at action level. The sent error code may be specified in the adjacent edit field.
Send transaction/action reply	If selected, the reply message can be edited in the pane below.

Send ContextReply	<p>If enabled, the reply contains the requested context parameters. To edit the sent context parameters, select this option and click Edit.</p> <p>When the Send transaction/action reply option is selected, the reply commands can be configured individually using the GUI. For each command reply sent, a property page (Notify Reply#n) is displayed and permits you to edit the following parameters:</p> <ul style="list-style-type: none"> • The reply terminationID, which can be any of the following: <ul style="list-style-type: none"> ■ ROOT : This denominates the ROOT termination. ■ \$: This denominates an arbitrary termination that is chosen by the device. ■ * : This denominates all terminations. ■ <AUTO-PHYSICAL> : This denominates the physical termination of the context associated with the current user. ■ <AUTO-RTP1> : This denominates the first RTP termination of the context associated with the current user. ■ <AUTO-RTP2> : This denominates the second RTP termination of the context associated with the current user. • A command-specific list of descriptors (the descriptors that can be contained in the current command) that are to be included in the command reply. When a descriptor is selected, the pane below displays an editable tree representation of the current descriptor. See Appendix C for more information on editing descriptors.
-------------------	--

Wait Notify: Parameters

The following table describes the execution parameters of the Wait Notify script function:

Name	Description
Delay Before Execution	<p>Delays the function execution by a duration that can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, Sleep 2000 in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms)

Timeout	<p>The time, in milliseconds (ms), that the script function waits for it to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled.</p> <p>It can be specified as follows:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Delay Between Digits, in milliseconds (ms) • PHONE_WAIT_TIME, in milliseconds (ms) • MGCP Timeout, in milliseconds (ms)
---------	--


Wait Notify: Output Settings

The following table describes the outputs parameters of the Wait Notify script function:

Output Name	Description
OK	The function completed successfully. The default resolution for this output is SUCCESS.
Timeout	The default resolution for this output is WARNING.
Error	The function has returned an internal error. The default resolution for this output is FAILED.


Wait SrvChange (MGC)

Waits for a transaction request with one or more ServiceChange commands for the specified timeout and then sends a transaction reply. If the corresponding H248 activity is configured with the **Use TransactionResponseAck** option, the script function also waits for a final acknowledgment. For description on the configuration parameters, see [Wait Notify: Rx Request](#), [Wait Notify: Tx Reply](#), [Wait Notify: Parameters](#), and [Wait Notify: Output Settings](#).

 **Note:** Wait-type script functions accept messages with H.248 protocol combinations that are not allowed in practice (such as, for example, Subtract from a NULL context, Add with a NULL context) or combinations that are not valid in the current state of the gateway (such as, for example, mixing unrelated terminations and contexts).

Wait Requests (MGC)

Waits for a transaction request with one or more specified H.248 commands for the specified timeout. If the transaction matches one of the expected commands, it sends the configured transaction reply and exits on the corresponding output. If a transaction request is received, but it doesn't match any of the templates, the 'Mismatched' output is used. If the corresponding H248 activity is configured with the **Use TransactionResponseAck** option, the script function also waits for a final acknowledgment. For a description of the expected command configuration, see [Wait Notify: Rx Request](#) and [Wait Notify: Tx Reply](#).

 **Note:** Wait-type script functions accept messages with H.248 protocol combinations that are not allowed in practice (such as, for example, Subtract from a NULL context, Add with a NULL context) or combinations that are not valid in the current state of the gateway (such as, for example, mixing unrelated terminations and contexts).

H.248 MGW Library

- [Notify](#)
- [SrvChange \(MGW\)](#)
- [Wait Add](#)
- [Wait Modify](#)
- [Wait Move](#)
- [Wait Subtract](#)
- [Wait AuditVal](#)
- [Wait AuditCap](#)
- [Wait SrvChange \(MGW\)](#)
- [Wait Requests \(MGW\)](#)

Notify


Sends a transaction request with one or more Notify commands and waits for the transaction reply (replies) for the specified timeout. A Notify command allows the Media Gateway to inform the Media Gateway Controller of the occurrence of events in the Media Gateway. For description on the configuration parameters, see [Add: Tx Request](#), [Add: Rx Reply](#), [Add: Parameters](#), and [Add: Output Settings](#).

SrvChange (MGW)

Sends a transaction request with one or more ServiceChange commands and waits the transaction reply for the specified timeout. A ServiceChange command allows the MG to notify the MGC that a termination or group of terminations is about to be taken out of service or has just been returned to service. ServiceChange commands are also used by the MG to announce its availability to a MGC (registration), and to notify the MGC of impending or completed restart of the MG. For description on the configuration parameters, see [Add: Tx Request](#), [Add: Rx Reply](#), [Add: Parameters](#), and [Add: Output Settings](#).


Wait Add

Waits for a transaction request with one or more Add commands for the specified timeout and then sends a transaction reply. An Add command adds a termination to a context. An Add command on the first termination in a context is used to create a context. For description on the configuration parameters, see [Wait Notify: Rx Request](#), [Wait Notify: Tx Reply](#), [Wait Notify: Parameters](#), and [Wait Notify: Output Settings](#).

 **Note:** Wait-type script functions accept messages with H.248 protocol combinations that are not allowed in practice (such as, for example, Subtract from a NULL context, Add with a NULL context) or combinations that are not valid in the current state of the gateway (such as, for example, mixing unrelated terminations and contexts).


Wait Modify

Waits for a transaction request with one or more Modify commands for the specified timeout and then sends a transaction reply. A Modify command modifies the properties, events and signals of a termination. For description on the configuration parameters, see [Wait Notify: Rx Request](#), [Wait Notify: Tx Reply](#), [Wait Notify: Parameters](#), and [Wait Notify: Output Settings](#).

 **Note:** Wait-type script functions accept messages with H.248 protocol combinations that are not allowed in practice (such as, for example, Subtract from a NULL context, Add with a NULL context) or combinations that are not valid in the current state of the gateway (such as, for example, mixing unrelated terminations and contexts).


Wait Move

Waits for a transaction request with one or more Move commands for the specified timeout and then sends a transaction reply. A Move command atomically moves a termination from one context to another. For description on the configuration parameters, see [Wait Notify: Rx Request](#), [Wait Notify: Tx Reply](#), [Wait Notify: Parameters](#), and [Wait Notify: Output Settings](#).

 **Note:** Wait-type script functions accept messages with H.248 protocol combinations that are not allowed in practice (such as, for example, Subtract from a NULL context, Add with a NULL context) or combinations that are not valid in the current state of the gateway (such as, for example, mixing unrelated terminations and contexts).

Wait Subtract

Waits for a transaction request with one or more Subtract commands for the specified timeout and then sends a transaction reply. A Subtract command disconnects a termination from its context and returns statistics on the termination's participation in the context. The Subtract command on the last termination in a context deletes the context. For description on the configuration parameters, see [Wait Notify: Rx Request](#), [Wait Notify: Tx Reply](#), [Wait Notify: Parameters](#), and [Wait Notify: Output Settings](#).

 **Note:** Wait-type script functions accept messages with H.248 protocol combinations that are not allowed in practice (such as, for example, Subtract from a NULL context, Add with a NULL context) or combinations that are not valid in the current state of the gateway (such as, for example, mixing unrelated terminations and contexts).

Wait AuditVal

Waits for a transaction request with one or more AuditValue commands for the specified timeout and then sends a transaction reply. An AuditValue command returns the current state of properties, events, signals and statistics of terminations. For description on the configuration parameters, see [Wait Notify: Rx Request](#), [Wait Notify: Tx Reply](#), [Wait Notify: Parameters](#), and [Wait Notify: Output Settings](#).

Note: Wait-type script functions accept messages with H.248 protocol combinations that are not allowed in practice (such as, for example, Subtract from a NULL context, Add with a NULL context) or combinations that are not valid in the current state of the gateway (such as, for example, mixing unrelated terminations and contexts).

Wait AuditCap

Waits for a transaction request with one or more AuditCapabilities commands for the specified timeout and then sends a transaction reply. An AuditCapabilities command returns all the possible values for the termination properties, events and signals allowed by the Media Gateway. For description on the configuration parameters, see [Wait Notify: Rx Request](#), [Wait Notify: Tx Reply](#), [Wait Notify: Parameters](#), and [Wait Notify: Output Settings](#).

Note: Wait-type script functions accept messages with H.248 protocol combinations that are not allowed in practice (such as, for example, Subtract from a NULL context, Add with a NULL context) or combinations that are not valid in the current state of the gateway (such as, for example, mixing unrelated terminations and contexts).

Wait SrvChange (MGW)

Waits for a transaction request with one or more ServiceChange commands for the specified timeout and then sends a transaction reply. A MGC may announce a handover to the MG by sending it a ServiceChange Command. The MGC may also use ServiceChange to instruct the MG to take a termination or group of terminations in or out of service. For description on the configuration parameters, see [Wait Notify: Rx Request](#), [Wait Notify: Tx Reply](#), [Wait Notify: Parameters](#), and [Wait Notify: Output Settings](#).

Wait Requests (MGW)

Waits for a transaction request with one or more H.248 commands for the specified timeout. If the transaction matches one of the expected requests, then it sends the configured transaction reply and exits on the corresponding output. If a transaction request is received but it doesn't match any of the templates, the 'Mismatched' output is used. For a description of the expected command configuration, see [Wait Requests \(MGC\)](#).

VoIP HTTP Functions Library

The VoIP HTTP Library includes the following script functions that emulate HTTP transactions.

HTTP Library

- [HTTP Transaction below](#)

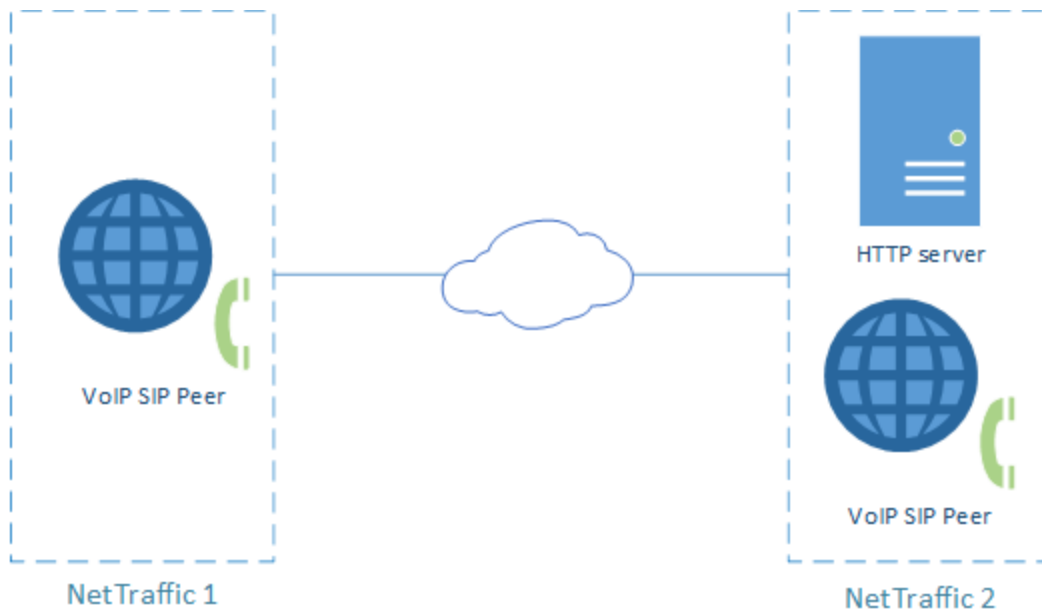
HTTP Transaction

The HTTP Transaction script function is available in the HTTP Library, which is under VoIP in the Workspace area on the Scenario tab.

HTTP Transaction is available for VoIPSIP activities only.

HTTP Transaction is intended for testing the integration of HTTP traffic within a SIP call.

You can use the Extract Variables function to extract variables from HTTP messages .See [VoIP SIP Functions Library on page 32](#)



Send Request tab

The controls on this tab configure the data that the transaction sends to the server.

Parameter	Description
Method	HTTP method that the transaction executes.
Destination	Destination of HTTP traffic from the transaction. <ul style="list-style-type: none"> • use activity settings: use the global destination defined on the HTTP tab. • use DNS or IP address: use the hostname or IP address configured in the Destination field.
Create new connection	Determines whether or not a transaction reuses a TCP connection. Enabled: The transaction always establishes a new TCP connection, even if the previously-used connection is still open. Disabled: If the previous TCP connection is still open, the transaction re-uses it.
Clear cookies	If enabled, any existing cookies are deleted and the HTTP message is sent without a Cookie header.
Page/Object	Web page or file object that the transaction acts on. Specify the full path to the page or object.

Parameter	Description
Headers	<p>HTTP headers sent in transaction.</p> <p>Headers are sent in the order they appear in the list. To move a header, select it and then drag it up or down in the list.</p> <p>To enable sending a header, check the checkbox next to the header. To disable sending a header, clear the checkbox.</p> <p>To add a header, click Add Header, then configure the header properties.</p> <p>To delete a header, select it, then click Remove Header.</p>
Body	<p>Body of the HTTP message.</p> <p>Click Body, then enter the text.</p>
Restore Defaults	Replaces the values configured on the tab with default values.

Receive Response tab

The controls on this tab configure how the transaction responds to the server.

Parameter	Description
Timeout	<p>Length of time, in milliseconds, that the transaction waits for a response from the server.</p> <p>If it does not receive a response within the timeout period, it logs an error and ends the transaction.</p>
Responses	<p>HTTP response codes that the transaction returns to the server. The transactions sends all the responses in the list, in the order they appear.</p> <p>To add a response, click Add Response and select it from the list.</p> <p>To delete a response, select it and click Remove.</p> <p>To move a response within the list, select it and click Up or Down.</p>
Restore Defaults	Replaces the values configured on the tab with default values.

Statistics

The following table lists the statistics reported by HTTP Transaction.

Statistic	Description
Request statistics	
HTTP Requests Sent	<p>Total number of Full-Requests sent successfully by the clients. This statistic is incremented after the final byte of the Full-Request has been sent.</p> <p>Full-Requests are described in Section 5 of RFC 1945.</p>

Statistic	Description
HTTP Requests Successful	Number of positive HTTP responses (2xx- and 3xx-range responses) received by the clients. The statistics show the number of requests for each URL.
HTTP Intermediate Responses Received (1xx)	Number of 100-series (Informational) responses received. 100-series responses indicate a provisional response, consisting only of the Status-Line and optional headers, and terminated by an empty line. Refer to RFC 2616, Section 10, for a full description.
HTTP Requests Successful (2xx)	Number of 200-series (Successful) responses received. 200-series responses indicate that the client's request was successfully received, understood, and accepted.
HTTP Requests Successful (3xx)	Number of 300-series (Redirection) responses received. 300-series responses indicate that further action needs to be taken by the user agent in order to fulfill the request.
HTTP Requests Successful (301)	Number of 301 (Moved Permanently) responses received. 301 responses indicate that the requested resource has been assigned a new permanent URI and any future references to this resource should use one of the returned URIs.
HTTP Requests Successful (302)	Number of 302 (Found) responses received. 302 responses indicate that the requested resource resides temporarily under a different URI.
HTTP Requests Successful (303)	Number of 303 (See Other) responses received. 303 responses indicate that the response to the request can be found under a different URI and should be retrieved using a GET method on that resource.
HTTP Requests Successful (307)	Number of 307 (Temporary Redirect) responses received. 307 responses indicate that the requested resource resides temporarily under a different URI.
HTTP Requests Failed	Number of HTTP requests that failed for any reason. The statistics show the number of requests for each URL (page).
HTTP Requests Failed (Write)	Number of HTTP requests that failed due to a socket write error. The statistics show the number of requests for each URL (page).
HTTP Requests Failed (Read)	Number of HTTP requests that failed due to a socket read error. The statistics show the number of requests for each URL (page).

Statistic	Description
HTTP Requests Failed (Bad Header)	Number of HTTP requests that failed due to a defective HTTP header. The statistics show the number of requests for each URL (page).
HTTP Requests Failed (4xx)	Number of 4xx-range responses received by the clients in response to an HTTP request. The statistics show the number of requests for each URL (page). 408 responses are counted separately by the HTTP Session Timeout (408) statistic and may or may not also be included in the HTTP Requests Failed (4xx) count. See the description of HTTP Session Timeout (408) for more information.
HTTP Requests Failed (400)	Bad Request. Number of requests that failed due to a syntax error in the URL. The statistics show the number of requests for each URL (page).
HTTP Requests Failed (401)	Unauthorized. Number of requests that failed due to because the server did not receive the correct user name or password from the browser. The statistics show the number of requests for each URL (page).
HTTP Requests Failed (403)	Forbidden. Number of requests that failed due to because the name or password supplied by the browser are incorrect. The statistics show the number of requests for each URL (page).
HTTP Requests Failed (404)	Not Found. Number of requests that failed because requested object is not stored on the server on the path supplied. The statistics show the number of requests for each URL (page).
HTTP Requests Failed (407)	Proxy Authentication Required. Number of requests that failed because access to the URL requires authentication with a proxy server.
HTTP Requests Failed (408)	Timeout. Number of requests that failed due to communications between the client and server taking too long. The statistics show the number of requests for each URL (page).
HTTP Requests Failed (4xx other)	Number of HTTP requests that failed for reasons other than a Bad Request (400), Unauthorized (401), Forbidden (403), Not Found (404), Proxy Authentication Required (407), or Timeout (408) error. The statistics show the number of requests for each URL (page).

Statistic	Description
HTTP Requests Failed (5xx)	Number of HTTP requests that failed due to lack of resources on the server (HTTP 500-series errors). This statistic is only incremented if the client had issued a request to the server before receiving the 5xx response. The statistics show the number of requests for each URL (page).
HTTP Requests Failed (505)	HTTP Version not Supported. Number of requests that failed because the server does not support the HTTP version used by the client. The statistics show the number of requests for each URL (page).
HTTP Requests Failed (5xx other)	Number of requests that failed for reasons other than an HTTP version mis-match (505). The statistics show the number of requests for each URL (page).
HTTP Requests Failed (other)	Number of requests that failed that could not be classified.
HTTP Requests Failed (Timeout)	Number of HTTP requests that failed because the clients did not receive a response within 600 seconds. The statistics show the number of requests for each URL (page).
HTTP Requests Failed (Aborted)	Number of HTTP requests that ended prematurely due to events outside HTTP or TCP. For example, if any HTTP requests are pending when the Ramp-Down period ends, those requests are aborted by IxLoad. The statistics show the number of requests for each URL (page).
Session statistics	
HTTP Session Timeouts (408)	<p>Number of HTTP 408 responses received. This statistic includes all 408 responses received regardless of whether they were received for a pending HTTP request or not.</p> <p>IxLoad counts 408 responses differently depending on whether or not a client has a pending HTTP request:</p> <ul style="list-style-type: none"> • If a client has an HTTP request pending and it receives a 408 response, IxLoad increments the HTTP Received 408, HTTP Requests Failed (4xx), and HTTP Requests Failed statistics. • If a client does not have an HTTP request pending and it receives a 408 response, IxLoad only increments the HTTP Received 408 statistic.
HTTP Sessions Rejected (503)	Service Unavailable. Number of HTTP sessions that could not be established due to lack of resources on the server.

Statistic	Description
Rate statistics	
HTTP Requests Sent/sec	Rate, per second, at which the client sent HTTP requests.
HTTP Requests Successful (2xx)/sec	Rate, per second, at which HTTP 2xx requests succeeded.
HTTP Requests Successful (3xx)/sec	Rate, per second, at which HTTP 3xx requests succeeded.
HTTP Requests Failed (4xx)/sec	Rate, per second, at which HTTP 4xx-series requests failed.
HTTP Requests Failed (5xx)/sec	Rate, per second, at which HTTP 5xx-series requests failed.
HTTP Requests Failed (other)/sec	Rate, per second, at which HTTP requests other than 2xx, 3xx, 4xx, or 5xx failed.
Connection and Transaction statistics	
HTTP Connections	Total number of connections established by the client.
HTTP Connection Attempts	Total number of connections attempted.
HTTP Transactions	Total number of transactions completed by the client.

VoIP MGCP Functions Library

The MGCP Test Library includes the following script functions:

- [Send NTFY](#)
- [Send DLCX \(GW\)](#)
- [Send RSIP](#)
- [Wait CRCX](#)
- [Wait DLCX \(GW\)](#)
- [Wait MDCX](#)
- [Wait RQNT](#)
- [Wait AUPE](#)
- [Wait AUCX](#)
- [Wait EPCF](#)
- [Wait Any Command \(GW\)](#)
- [Wait Command \(GW\)](#)

- [Send RQNT](#)
- [Send CRCX](#)
- [Send DLCX \(CA\)](#)
- [Send MDCX](#)
- [Send AUCX](#)
- [Send AUEP](#)
- [Send EPCF](#)
- [Wait NTFY](#)
- [Wait DLCX \(CA\)](#)
- [Wait Command \(CA\)](#)
- [Wait Any Command \(CA\)](#)
- [Wait RSIP](#)

Overview

Each script function implements an MGCP transaction, that is, an MGCP command followed by an expected response.

A script function typically represents a sent MGCP command and the expected response message (for Send type functions), the corresponding transaction being closed when a matching response message is received.

MGCP MGW Functions

The following script functions are available:

Send NTFY

The MGCP Notify command is used by a MGW to send notifications of the observed events when a triggering event occurs. The Send Notify script function implements the transaction initiated by a Notify command.

Send Notify Properties: Tx Command

This page specifies the Notify command parameters. See Tx Command Page in Appendix D.

Send Notify Properties: Rx Response

This page specifies the awaited response message following the sending of a Notify command. See Rx Response Page in Appendix D.

Send Notify Properties: Parameters

The following table describes the Send Notify function parameters:

Name	Description
------	-------------

Delay before execution	<p>Delays the function execution by a value specified as:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms)
Transaction timeout	<p>The time, in milliseconds (ms), that the script function waits for it to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled.</p> <p>It can be configured as either of the following:</p> <ul style="list-style-type: none"> • A static user-defined value that is entered into the field. • A random value between two user-specified values. • The timeout value configured at the activity level (Use activity settings option).

Send Notify Properties: Output Settings

The following table describes the Send Notify output parameters:

Name	Description
200 OK	The Send Notify function was executed successfully. The default resolution for this output is SUCCESS.
Timeout	A timeout occurred while executing the Send Notify function. The default resolution for this output is WARNING.
Error	The Send Notify function execution has returned an error. The default resolution for this output is FAILED.

Send DLCX (GW)

In some rare circumstances, a MGW may have to clear a connection, for example because it has lost the resource associated with the connection, or because it has detected that the endpoint no longer is capable or willing to send or receive media. In such situations, the MGW may then terminate the connection by using an MGCP DeleteConnection command.

The Send DLCX script function implements the transaction initiated by a sent DeleteConnection command.

Send DLCX Properties: Tx Command

This page specifies the DLCX command parameters. See Tx Command Page in Appendix D.

Send Notify Properties: Rx Response

This page specifies the awaited response message following the sending of a DLCX command. See Rx Response Page in Appendix D.

Send DLCX Properties: Parameters

This page specifies the function execution parameters described in [Send Notify Properties: Parameters](#).

Send DLCX Properties: Output Settings

The following table describes the Send DLCX Output parameters:

Name	Description
250 Connection Deleted	The Send DLCX function was executed successfully. The default resolution for this output is SUCCESS.
Timeout	A timeout occurred while executing the Send DLCX function. The default resolution for this output is WARNING.
Error	The Send DLCX function execution has returned an error. The default resolution for this output is FAILED.

Send RSIP

The Send RSIP script function implements the transaction initiated by a RestartInProgress (RSIP) command sent from the MGW to the CA.

Send RSIP Properties: Tx Command

This page specifies the RSIP command parameters. See Tx Command Page in Appendix D.

Send RSIP Properties: Rx Response

This page specifies the awaited response message following the sending of a RSIP command. See Rx Response Page in Appendix D.

Send RSIP Properties: Parameters

This page specifies the function execution parameters described in [Send Notify Properties: Parameters](#).

Send RSIP Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send DLCX Properties: Output Settings](#).

Wait CRCX

The Wait CRCX script function implements the transaction initiated by the receiving of a CreateConnection (CRCX) command, followed by the sending of a response to this command.

Wait CRCX Properties: Rx Command

This page specifies the parameters of a received CRCX command. See Rx Command Page in Appendix D.

Wait CRCX Properties: Tx Response

This page specifies the generated response message following the receiving of a CRCX command. See Tx Response Page in Appendix D.

Wait CRCX Properties: Parameters

This page specifies the function execution parameters described in [Wait Command Properties: Parameters](#).

Wait CRCX Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Wait DLCX (GW)

The Wait DLCX script function implements the transaction initiated by the receiving of a DeleteConnection (CRCX) command, followed by the sending of a response to this command.

Wait DLCX Properties: Rx Command

This page specifies the parameters of a received DLCX command. See Rx Command Page in Appendix D.

Wait DLCX Properties: Tx Response

This page specifies the generated response message following the receiving of a DLCX command. See Tx Response Page in Appendix D.

Wait DLCX Properties: Parameters

This page specifies the function execution parameters, the same as those described in [Wait Command Properties: Parameters](#).

Wait DLCX Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Wait MDCX

The Wait MDCX script function implements the transaction initiated by a the receiving of a ModifyConnection (CRCX) command, followed by the sending of a response to this command.

Wait MDCX Properties: Rx Command

This page specifies the parameters of a received MDCX command. See Rx Command Page in Appendix D.

Wait MDCX Properties: Tx Response

This page specifies the generated response message following the receiving of a MDCX command. See Tx Response Page in Appendix D.

Wait MDCX Properties: Parameters

This page specifies the function execution parameters, the same as those described in [Wait Command Properties: Parameters](#).

Wait MDCX Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Wait RQNT

The Wait RQNT script function implements the transaction initiated by the receiving of a RequestNotify (RQNT) command, followed by the sending of a response to this command.

Wait RQNT Properties: Rx Command

This page specifies the parameters of a received RQNT command. See Rx Command Page in Appendix D.

Wait RQNT Properties: Tx Response

This page specifies the generated response message following the receiving of a RQNT command. See Tx Response Page in Appendix D.

Wait RQNT Properties: Parameters

This page specifies the function execution parameters, the same as those described in [Wait Command Properties: Parameters](#).

Wait RQNT Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Wait AUEP

The Wait AUEP script function implements the transaction initiated by the receiving of an AuditEndpoint (AUEP) command, followed by the sending of a response to this command.

Wait AUEP Properties: Rx Command

This page specifies the parameters of a received AUEP command. See Rx Command Page in Appendix D.

Wait AUEP Properties: Tx Response

This page specifies the generated response message following the receiving of an AUEP command. See Tx Response Page in Appendix D.

Wait AUEP Properties: Parameters

This page specifies the function execution parameters, the same as those described in [Wait Command Properties: Parameters](#).

Wait AUEP Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Wait AUCX

The Wait AUCX script function implements the transaction initiated by the receiving of an AuditConnection (AUCX) command, followed by the sending of a response to this command.

Wait AUCX Properties: Rx Command

This page specifies the parameters of a received AUCX command. See Rx Command Page in Appendix D.

Wait AUCX Properties: Tx Response

This page specifies the generated response message following the receiving of an AUCX command. See Tx Response Page in Appendix D.

Wait AUCX Properties: Parameters

This page specifies the function execution parameters, the same as those described in [Wait Command Properties: Parameters](#).

Wait AUCX Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Wait EPCF

The Wait EPCF script function implements the transaction initiated by the receiving of an EndpointConfiguration (EPCF) command, followed by the sending of a response to this command.

Wait EPCF Properties: Rx Command

This page specifies the parameters of a received EPCF command. See Rx Command Page in Appendix D.

Wait EPCF Properties: Tx Response

This page specifies the generated response message following the receiving of an EPCF command. See Tx Response Page in Appendix D.

Wait EPCF Properties: Parameters

This page specifies the function execution parameters, the same as those described in [Wait Command Properties: Parameters](#).

Wait EPCF Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Wait Any Command (GW)

The Wait Any Command (GW) command waits for any MGCP command with a user-specified set of parameters.

Wait Any Command (GW) Properties: Rx Command

This page specifies the parameters of a received MGCP command. See Rx Command Page in Appendix D.

Wait Any Command (GW) Properties: Tx Response

This page specifies the generated response message following the receiving of a MGCP command. See Tx Response Page in Appendix D.

Wait Any Command (GW) Properties: Parameters

The following table describes the Wait Any function parameters:

Name	Description
Delay before execution	Delays the function execution by a value specified as: <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms)
Timeout	The time, in milliseconds (ms), that the script function waits for it to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled. <ul style="list-style-type: none"> • It can be configured as either of the following: <ul style="list-style-type: none"> • A static user-defined value that is entered into the field. • A random value between two user-specified values • The timeout value configured at the activity level (Use activity settings option).
Transaction delay	The time, in milliseconds (ms), that the script function waits before executing. It can be configured as either of the following: <ul style="list-style-type: none"> • A static user-defined value that is entered into the field. • A random value between two user-specified values.

Wait Any Command (GW) Properties: Output Settings

The following table describes the Wait Any Output parameters:

Name	Description
OK	The Wait Any function was executed successfully. The default resolution for this output is SUCCESS.




Timeout	A timeout occurred while executing the Wait Any function. The default resolution for this output is WARNING.
Error	The Wait Any function execution has returned an error. The default resolution for this output is FAILED.

Wait Command (GW)

The Wait Command (GW) function specifies one or more MGCP commands awaited by the MGW. If any of these commands is received, the user-configured response to the command is sent and, if executed successfully, the function exits on the output identifying the matched command.

Wait Command Properties: Templates

The following table describes the request parameters of the Wait Command (GW) script function:

Name	Description																
	<p>A list of commands to be matched by the MGW. If any of the received commands is matched, the function exits on the corresponding output. For example, considering we have a configured list such as the following, if an MDCX-type command with matching parameters is received, then the Wait Command function sends the 200 Ok response code and exits on the MDCX1 output that was automatically created when adding the command to the list.</p> <table border="1"> <thead> <tr> <th>Command Type</th> <th>Output Name</th> <th>Response Code</th> <th>Ignore SDP</th> </tr> </thead> <tbody> <tr> <td>CRCX</td> <td>CRCX1</td> <td>auto</td> <td>no</td> </tr> <tr> <td>MDCX</td> <td>MDCX1</td> <td>200</td> <td>no</td> </tr> <tr> <td>AUCX</td> <td>AUCX1</td> <td>auto</td> <td>N/A</td> </tr> </tbody> </table>	Command Type	Output Name	Response Code	Ignore SDP	CRCX	CRCX1	auto	no	MDCX	MDCX1	200	no	AUCX	AUCX1	auto	N/A
Command Type	Output Name	Response Code	Ignore SDP														
CRCX	CRCX1	auto	no														
MDCX	MDCX1	200	no														
AUCX	AUCX1	auto	N/A														
	Click Add to add a new command to be matched by the MGW. For each added command, the awaited MGCP command and parameters, as well as the response can be configured as described in Appendix D.																
	Click Delete to delete the currently selected command from the list of awaited commands. You cannot delete the last remaining command.																
	Click Edit to view a window that permits you to edit the command parameters and the command response.																

Wait Command Properties: Parameters

The following table describes the Wait Command function parameters:

Name	Description
Delay before execution	<p>Delays the function execution by a value specified as:</p> <ul style="list-style-type: none"> Static Expression, in milliseconds (ms) Random Between Expressions, in milliseconds (ms)

Timeout	<p>The time, in milliseconds (ms), that the script function waits for it to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled.</p> <ul style="list-style-type: none"> • It can be configured as either of the following: <ul style="list-style-type: none"> • A static user-defined value that is entered into the field. • A random value between two user-specified values • The timeout value configured at the activity level (Use activity settings option).
Transaction delay	<p>The time, in milliseconds (ms), that the script function waits before executing. It can be configured as either of the following:</p> <ul style="list-style-type: none"> • A static user-defined value that is entered into the field. • A random value between two user-specified values.

Wait Command Properties: Output Settings

In addition to the common Timeout and Error outputs available for all script functions, the Wait Command function provides additional outputs for all awaited messages configured in the function. These additional outputs are automatically created when adding a command to the list of commands to be matched. For example, assuming we had configured the list of commands as shown in the following image, the additional MDCX1 and AUCX1 outputs would become available in the script function.

Command Type	Output Name	Response Code	Ignore SDP
CRCX	CRCX1	auto	no
MDCX	MDCX1	200	no
AUCX	AUCX1	auto	N/A

MGCP CA Functions

The following functions are available:

Send RQNT

The Send RQNT script function implements the transaction initiated by a RequestNotify (RQNT) command sent by the CA.

Send RQNT Properties: Tx Command

This page specifies the RQNT command parameters. See Tx Command Page in Appendix D.

Send RQNT Properties: Rx Response

This page specifies the awaited response message following the sending of an RQNT command. See Rx Response Page in Appendix D.

Send RQNT Properties: Parameters

This page specifies the function execution parameters described in [Send Notify Properties: Parameters](#).

Send RQNT Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Send CRCX

The Send CRCX script function implements the transaction initiated by a CreateConnection (CRCX) command sent by the CA.

Send CRCX Properties: Tx Command

This page specifies the CRCX command parameters. See Tx Command Page in Appendix D.

Send CRCX Properties: Rx Response

This page specifies the awaited response message following the sending of an CRCX command. See Rx Response Page in Appendix D.

Send CRCX Properties: Parameters

This page specifies the function execution parameters described in [Send Notify Properties: Parameters](#).

Send CRCX Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Send DLCX (CA)

The Send DLCX script function implements the transaction initiated by a DeleteConnection (DLCX) command sent by the CA.

Send DLCX Properties: Tx Command

This page specifies the DLCX command parameters. See Tx Command Page in Appendix D.

Send DLCX Properties: Rx Response

This page specifies the awaited response message following the sending of an DLCX command. See Rx Response Page in Appendix D.

Send DLCX Properties: Parameters

This page specifies the function execution parameters described in [Send Notify Properties: Parameters](#).

Send DLCX Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Send MDCX

The Send MDCX script function implements the transaction initiated by a ModifyConnection (MDCX) command sent by the CA.

Send MDCX Properties: Tx Command

This page specifies the MDCX command parameters. See Tx Command Page in Appendix D.

Send MDCX Properties: Rx Response

This page specifies the awaited response message following the sending of an MDCX command. See Rx Response Page in Appendix D.

Send MDCX Properties: Parameters

This page specifies the function execution parameters described in [Send Notify Properties: Parameters](#).

Send MDCX Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Send AUCX

The Send AUCX script function implements the transaction initiated by an AuditConnection (AUCX) command sent by the CA.

Send AUCX Properties: Tx Command

This page specifies the AUCX command parameters. See Tx Command Page in Appendix D.

Send AUCX Properties: Rx Response

This page specifies the awaited response message following the sending of an AUCX command. See Rx Response Page in Appendix D.

Send AUCX Properties: Parameters

This page specifies the function execution parameters described in [Send Notify Properties: Parameters](#).

Send AUCX Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Send AUEP

The Send AUEP script function implements the transaction initiated by an Audit Connection (AUEP) command sent by the CA.

Send AUEP Properties: Tx Command

This page specifies the AUEP command parameters. See Tx Command Page in Appendix D.

Send AUEP Properties: Rx Response

This page specifies the awaited response message following the sending of an AUEP command. See Rx Response Page in Appendix D.

Send AUEP Properties: Parameters

This page specifies the function execution parameters described in [Send Notify Properties: Parameters](#).

Send AUEP Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Send EPCF

The Send EPCF script function implements the transaction initiated by a Endpoint Configuration (EPCF) command sent by the CA.

Send EPCF Properties: Tx Command

This page specifies the EPCF command parameters. See Tx Command Page in Appendix D.

Send EPCF Properties: Rx Response

This page specifies the awaited response message following the sending of an EPCF command. See Rx Response Page in Appendix D.

Send EPCF Properties: Parameters

This page specifies the function execution parameters described in [Send Notify Properties: Parameters](#).

Send EPCF Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Wait NTFY

The Wait NTFY script function implements the transaction initiated by the receiving of an MGCP Notify command sent by an MGW.

Wait NTFY Properties: Rx Command

This page specifies the parameters of a received NTFY command. See Rx Command Page in Appendix D.

Wait NTFY Properties: Tx Response

This page specifies the generated response message following the receiving of an NTFY command. See Tx Response Page in Appendix D.

Wait NTFY Properties: Parameters

This page specifies the function execution parameters, the same as those described in [Wait Command Properties: Parameters](#).

Wait NTFY Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Wait DLCX (CA)

The Wait DLCX script function implements the transaction initiated by the receiving at the CA of an MGCP DeleteConnection (DCLX) command sent by an MGW.

Wait DLCX Properties: Rx Command

This page specifies the parameters of a received DLCX command. See Rx Command Page in Appendix D.

Wait DLCX Properties: Tx Response

This page specifies the generated response message following the receiving of an DLCX command. See Tx Response Page in Appendix D.

Wait DLCX Properties: Parameters

This page specifies the function execution parameters, the same as those described in [Wait Command Properties: Parameters](#).

Wait DLCX Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Wait Command (CA)

The Wait Command (CA) script function specifies one or more MGCP commands awaited by the CA. If any of these commands is received, the user-configured response to the command is sent and, if executed successfully, the function exits on the output identifying the matched command. See [Wait Any Command \(GW\)](#).

Wait Any Command (CA)

The Wait Any Command (CA) command waits for any MGCP command with a user-specified set of parameters. See [Wait Command \(GW\)](#).

Wait RSIP

The Wait RSIP script function implements the transaction initiated by the receiving at the CA of an MGCP RestartInProgress (RSIP) command sent by an MGW.

Wait RSIP Properties: Rx Command

This page specifies the parameters of a received RSIP command. See Rx Command Page in Appendix D.

Wait RSIP Properties: Tx Response

This page specifies the generated response message following the receiving of an RSIP command. See Tx Response Page in Appendix D.

Wait RSIP Properties: Parameters

This page specifies the function execution parameters, the same as those described in [Wait Command Properties: Parameters](#).

Wait RSIP Properties: Output Settings

This page specifies the function output parameters, the same as those described in [Send Notify Properties: Output Settings](#).

Digital T1/E1 Functions Library

The Digital T1/E1 Test Library includes the following script functions:

- [Make Call](#)
- [Receive Call](#)
- [End Call](#)
- [Path Confirmation](#)
- [Talk](#)
- [Listen](#)
- [Voice Session](#)
- [Generate DTMF](#)
- [Detect DTMF](#)
- [Generate MF](#)
- [Detect MF](#)
- [Generate Tone](#)
- [Wait for Tone](#)

Make Call

This script function initiates a call to the specified destination.

Make Call Properties: Parameters

The following table describes the Make Call function parameters:

Name	Description
Delay before execution	<p>Delays the function execution by a value specified as:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 2000, in milliseconds (ms)
No Answer Timeout	<p>The time, in milliseconds (ms), that the script function waits for the function to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled.</p> <p>This parameter can be specified as either of the following:</p> <ul style="list-style-type: none"> • A user-defined value that is entered into the field. • As the Global value, in which case the value is used configured in the IxLoad Global Settings window.

Destination Number	The destination number for the call originated by the PSTNDigital activity, which can be specified as either of the following: <ul style="list-style-type: none"> • Use Dest. Phone Value: The destination number is taken from the activity's dial plan configuration. • Custom: A user-configured value.
--------------------	--

Make Call Properties: Output Settings

The following table describes the Make Call output parameters:

Properties	Description
Output Settings	
Connected	A call was successfully established. The default resolution for this output is SUCCESS.
Timeout	A timeout occurred while trying to establish the call. The default resolution for this output is WARNING.
Disconnected	On the established connection, the remote party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.
Error	The function has returned an error following an inexistent call handle, inexistent line, or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

Receive Call

This script function answers an incoming call.

Receive Call Properties: Parameters

The following table describes the Receive Call function parameters:

Name	Description
Delay before answer	Delays the function execution by a value specified as: <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 2000, in milliseconds (ms)

Wait for call	<p>The time, in milliseconds (ms), that the script function waits for it to execute. If this time interval terminates without the function being executed, the Timeout function output is enabled.</p> <p>This parameter can be specified as either of the following:</p> <ul style="list-style-type: none"> • A user-defined value that is entered into the field. • As the Global value, in which case the value is used configured in the IxLoad Global Settings window.
Reject Call	<p>When a call is rejected by the Receive Call function, this parameter can be used for providing a call reject reason. This parameter can be configured to one of the predefined reasons, or to a Global value, in which case the used value is that configured in the IxLoad Global Settings window.</p>

Receive Call Properties: Output Settings

The following table describes the Receive Call output parameters:

Properties	Description
Output Settings	
Connected	A call was successfully received. The default resolution for this output is SUCCESS.
Timeout	A timeout occurred while receiving the call. The default resolution for this output is WARNING.
Disconnected	On the established connection, the remote party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.
Error	The function has returned an error following an inexistent call handle, inexistent line, or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

End Call

This script function terminates an established call.

End Call Properties: Parameters

The following table describes the End Call function parameters:

Name	Description
------	-------------

Delay before execution	<p>Delays the function execution by a value specified as:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 2000, in milliseconds (ms)
Wait other party to disconnect	<p>If selected, waits for the configured amount of time (in milliseconds) for the other party to disconnect the call. When this option is not selected, the End Call script function initiates the call disconnection itself.</p> <p>When this option is selected, the Wait duration parameter can be specified as either of the following:</p> <ul style="list-style-type: none"> • A user-defined value that is entered into the field. • As the Global value, in which case the value is used configured in the Global Settings window.

End Call Properties: Output Settings

The following table describes the End Call output parameters:

Properties	Description
Output Settings	
Ok	A call was successfully ended. The default resolution for this output is SUCCESS.
Error	The function has returned an error following an inexistent call handle, inexistent line, or erroneous parameters evaluation condition. The default resolution for this output is FAILED.

Path Confirmation


This script function executes a Path Confirmation sequence, wherein the path confirmation initiator sends a specific digit (DTMF/MF/tone) sequence and then waits to receive another digit (DTMF/MF/Tone) sequence from the remote party.

Path Confirmation Properties: Parameters

The following table describes the Path Confirmation function parameters:

Name	Description
------	-------------

<p>Delay before execution</p>	<p>Delays the function execution by a value specified as:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 2000, in milliseconds (ms)
<p>Path Confirmation Method</p>	<p>Selects the operating mode as either of the following:</p> <ul style="list-style-type: none"> • Simplex (default): Transmission and reception of the path confirmation sequence is done in an alternating manner. • Duplex: Transmission and reception of the path confirmation sequence is done in an alternating manner. • Synchronized Duplex (VoIP Style): Transmission and reception of the path confirmation sequence is done in both directions simultaneously.
<p>Specified Digit Sequence of...</p>	<p>Selects the mode – DTMFs, MFs, or tones – in which the path confirmation operation is performed:</p> <ul style="list-style-type: none"> • Specified Digit Sequence of DTMFs - The path confirmation initiator sends the specified DTMF digits and expects to receive the same DMTF digits. The path confirmation receiver expects the specified DTMF digits and then sends them back. • Specified Digit Sequence of MFs - The path confirmation initiator sends the specified MF digits and expects to receive the same MF digits. The path confirmation receiver expects the specified MF digits and then sends them back. • Specified Sequence of Custom Tones - The path confirmation initiator sends the specified custom tone sequence and expects to receive the same sequence. The path confirmation receiver expects the specified custom tone sequence and then sends it back.
<p>Time Guard Delay</p>	<p>This parameter specifies the time used as guard for path confirmation that supports the separation of Tx/Rx paths in simplex mode, and separation of phases of sequences for duplex modes. It can be a value in ms with valid values between 0 (default) and 60000 ms).</p>
<p>Start Mode</p>	<p>Specifies how the channels running this script function start the path confirmation sequence:</p> <ul style="list-style-type: none"> • Initiate - the channel(s) executing this script function first send the path confirmation digits and then wait, and so on (alternating). • Wait for - the channel(s) executing this script function first wait for the path confirmation digits and then send the digits, and so on (alternating). • Auto - the channel(s) executing this script function rely on signaling to decide if they wait for or send the digit sequence. The path confirmation initiator is the party that receives the call. This is the default option.

 **Note:** When a DTMF/MF/Tone sequence is specified using expressions, you must ensure that every two pair channels evaluate the expressions to the same value, otherwise the path confirmation sequence fails; the path confirmation initiator/receiver sends and expects the evaluated expression (interpreted as a digit sequence).

Path Confirmation Properties: Tone Detection/Generation

The following table describes the Path Confirmation Tone Detection/Generation parameters:

Name	Description
DTMF/MF Generation	<p>Specifies the following DTMF/MF generation parameters using custom values, or by applying the values specified in the Global Settings > T1/E1 > Voice window when the Use values from Global Settings option is selected:</p> <ul style="list-style-type: none"> • Tone Duration: The time (in ms) required for a single tone (DTMF/MF/Custom Tone) to be generated. The range of values is 50 through 10000 ms. The default value is 200 ms. • Inter Tone Interval: The maximum amount of time (in ms) between two consecutive generated DTMFs/ MFs/Custom Tones. The range of values is 50 to 10000 ms. The default value is 200 ms. • Signal Power: The attenuation (in dBm) of the DTMF/ MF. The minimum attenuation is -54 dBm and the maximum is -3 dBm. The default value is -10 dBm.
DTMF/MF/Tone Detection	<p>Specifies the following DTMF/MF/Tone detection parameters using custom values, or by applying the values specified in the Global Settings > T1/E1 > Voice window when the Use values from Global Settings option is selected:</p> <ul style="list-style-type: none"> • Maximum delay between tones: The maximum amount of time (in ms) allowed between consecutive digits (DTMFs/MFs/custom tones) for a proper detection. After this period elapses, the function exits on the Timeout output. The range of values is 0 to 65520 ms. The default value is 500 ms. • First Tone Timeout: The time (in ms) allowed for detecting the first digit (DTMF/MF/custom tone). After this period elapses, the function exits on Timeout output. The range of values is 0 to 65520 ms. The default value is 500 ms.

Path Confirmation Properties: Advanced Settings

The following table describes the Path Confirmation Advanced Settings parameters:

Name	Description
------	-------------

Playback	<p>Specifies the DTMF/MF/tone playback mode using either of the options below, or by applying the values specified in the Global Settings > T1/E1 > Voice window (for the Use Global Setting option enabled):</p> <ul style="list-style-type: none"> • Play: Specifies the number of times to play. • Repeat Continuously for: Specify the period of time to play. The default value is 1000 seconds. • Use Talk Time (for all objectives except Channels): Plays the DTMF/MF/Tone sequence for the duration of the Talk Time call parameter.
----------	--

Path Confirmation Properties: Output Settings

The following table describes the Path Confirmation Output Settings parameters:

Properties	Description
Output Settings	
Connected	The path confirmation sequence was correctly executed for the specified number of times or duration. The default resolution for this output is SUCCESS.
Timeout	The path confirmation sequence failed because of a timeout (for example, DTMF/MF/Tone detection timeout). The default resolution for this output is WARNING.
Disconnected	On the established connection, the remote party dropped the call by sending a disconnect indication. The default resolution for this output is WARNING.
Error	The function has returned an error. The default resolution for this output is FAILED.


Talk

This script function plays back a wave file or a QoV clip from the IxLoad Resource Pool.

Talk Properties: Parameters

The following table describes the Talk function parameters:

Name	Description
Delay Before Execution	<p>Delays the function execution by a value specified as:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 2000, in milliseconds (ms)

Playback Settings	<p>When the Overwrite playback activity setting option is selected, the following configured playback settings override the activity-level settings:</p> <ul style="list-style-type: none"> • Clip: Selects a wave file or a QoV clip or to be played back by the script function. Clicking the  opens a window that permits you to select the audio file from the IxLoad Resource Manager. • Output level: In the case of QoV clips, this specifies an output level of -20, -25, -30, or -35 dBm. • Play: Specifies the number of times to play. • Repeat continuously for: Specifies a period of time to play. The default value is 1000 seconds. • Use Talk Time (for all objectives except Channels): Plays the wave file for the duration of the Talk Time call parameter. • Use Global settings: The playback values are those specified in the Global Settings > T1/E1 > Voice window.
-------------------	--

Talk Properties: Advanced Settings

The following table describes the Talk function Advanced parameters:

Name	Description
Stop playback on first detected DTMF	If selected, the clip playback is stopped when a specified DTMF is detected.

Talk Properties: Output Settings

The following table describes the Talk function Output parameters:

Properties	Description
Output Settings	
Ok	The function was executed correctly executed for the specified number of times or duration. The default resolution for this output is SUCCESS.
Error	The function has returned an error. The default resolution for this output is FAILED.

Listen

This script function allows the recording of a wave file or QoV clip for a specified duration.

Listen Properties: Parameters

The following table describes the Listen function parameters:

Name	Description
------	-------------

Delay Before Execution	<p>Delays the function execution by a value specified as:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 2000, in milliseconds (ms)
Listen Settings	<p>Specifies the following recording settings:</p> <ul style="list-style-type: none"> • Data Format: The data format can be Mu-Law, A-Law, PCM, or as specified by the Global Settings > T1/E1 > Voice window. Default is Global. • Sampling Rate: The sampling rate value can be 6000 Hz, 8000 Hz, 11025 Hz, or as specified by the Global Settings > T1/E1 > Voice window. • Bits per sample: The sample size can be 8 bit, 16 bit, or as specified by the Global Settings > T1/E1 > Voice window. <p>When the Overwrite playback activity setting option is selected, the following configured playback settings override the activity-level settings:</p> <ul style="list-style-type: none"> • Listen Duration: If selected, the recording duration specified as a value or a formula (expression). The maximum duration value is 10 minutes. By default, this option is selected and configured to a 10 s value. • Use Talk Time (for all objectives except Channels): If selected, the recording duration is set to the value of the Talk Time call parameter.
Perform QoV measurements	<p>If selected, P.862 PESQ and P56 QoV scores computation is performed for the reference audio file specified in the Clip field. The clip's volume is specified by the adjacent Output level field.</p>

Listen Properties: Advanced Settings

The following table describes the Voice Session function Advanced Playback parameters:

Name	Description
Listen Terminate Conditions - Stop recording on first detected DTMF	If selected, the clip recording is stopped when a specified DTMF is detected.

Listen Properties: Output Settings

The following table describes the Listen function Output Settings parameters:

Properties	Description
Output Settings	

Ok	The function was executed correctly executed for the specified number of times or duration. The default resolution for this output is SUCCESS.
Error	The function has returned an error. The default resolution for this output is FAILED.

Voice Session

This script function plays back a wave file or QoV clip and records an audio file at the same time.

Voice Session Properties: Talk Parameters

The Voice Session Talk parameters are the same as those from [Talk Properties: Parameters](#).

Voice Session Properties: Listen Parameters

The Voice Session Listen parameters are the same as those from [Listen Properties: Parameters](#).

Voice Session Properties: Advanced Settings

The following table describes the Voice Session function Advanced Playback parameters.

Name	Description
Stop playback on first detected DTMF	If selected, the audio file playback is stopped when a specified DTMF is detected.
Stop recording on first detected DTMF	If selected, the audio file recording is stopped when a specified DTMF is detected.

Voice Session Properties: Output Settings

The following table describes the Voice Session function Output Settings parameters:

Properties	Description
Output Settings	
Ok	The function was executed correctly executed for the specified number of times or duration. The default resolution for this output is SUCCESS.
Error	The function has returned an error. The default resolution for this output is FAILED.

Generate DTMF

This script function generates a sequence of Dual Tone Multiple Frequency (DTMF) signals. The sequence of tones can have up to 31 DTMFs and the tone length and the inter-tone interval can be specified. The sequence may contain any combination of standard tones (that is, '1,' '2,' '3,' '4,' '5,' '6,' '7,' '8,' '9,' '0,' '#,' '*', 'a,' 'b,' 'c,' 'd.')

Generate DTMF Properties: Parameters

The following table describes the Generate DTMF function parameters:

Name	Description
Delay Before Execution	Delays the function execution by a value specified as: <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 2000, in milliseconds (ms)
Tone duration	The time length of a DTMF tone in the tone sequence. The range of values is Global, or 50 to 10000 ms. The default value is 200 ms.
InterDigit Delay	The delay between 2 consecutive DTMF tones in the tone sequence, including the silence time. The range is Global, or 50 to 10000 ms. The default value is 200 ms.
Signal power	The amplitude of the generated DTMF tones, measured in decibels (dBm). The range of values can be Global ,or from -54 to -3 dBm. The default value is -3 dBm.
DTMF Sequence	The sequence of DTMF tones to be generated, which can be local or can be chosen from the DTMF Pool. The default is the '12345' DTMF sequence.

Generate DTMF Properties: Advanced Settings

The following table describes the Generate DTMF Advanced Settings parameters:

Name	Description
Playback	Specifies the DTMF playback mode using either of the options below, or by applying the values specified in the Global Settings > T1/E1 > Voice window (for the Use Global Setting option enabled): <ul style="list-style-type: none"> • Play: Specifies the number of times to play. • Repeat Continuously for: Specify the period of time to play. The default value is 1000 seconds. • Use Talk Time (All objectives except Channels): Plays the DTMF sequence for the duration of the Talk Time call parameter. • Use Global Settings: If selected, the global settings configured in the Global Settings > T1/E1 > Voice page are used.

Generate DTMF Properties: Output Settings

The following table describes the Generate DTMF function Output Settings parameters:

Properties	Description
------------	-------------

Output Settings	
Ok	The function was executed correctly executed for the specified number of times or duration. The default resolution for this output is SUCCESS.
Error	The function has returned an error. The default resolution for this output is FAILED.

Detect DTMF

This script function is used to detect a sequence of DTMF signals. The sequence can have up to 31 tones and the inter-tone interval can be specified. The intertone interval includes the silence time between two consecutive tones. The expected sequence to be received may contain any of the standard tones (that is, '1,' '2,' '3,' '4,' '5,' '6,' '7,' '8,' '9,' '0,' '#,' '*', 'a,' 'b,' 'c,' 'd.')

Detect DTMF Properties: Parameters

The following table describes the Detect DTMF function parameters:

Name	Description
Delay Before Execution	<p>Delays the function execution by a value specified as:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 2000, in milliseconds (ms)
DTMF Detection Settings	<p>Specifies the following detection parameters:</p> <ul style="list-style-type: none"> • Detect Continuously for: Detects all the DTMFs arrived in the specified period of time (measured in seconds, minutes or hours). It can be a value or a formula (expression). Maximum duration value is 10 minutes. The default value is 1000 seconds. • Detect Exactly: Detects the specified number of DTMFs. The default value is 6 DTMFs. • Use Talk Time (all objectives except Channels): Performs detection for the duration of the Talk Time call parameter. • Detect DTMF Sequence: Detects the expected sequence (specified by selecting an entry in the DTMF Sequence pool items, or a Local Sequence). The default is 12345 DTMF string.

Terminate Conditions	<p>Specifies the termination condition of the DTMF detection as either of the following:</p> <ul style="list-style-type: none"> Maximum delay between tones: If selected, the DTMF detection is terminated when tones arrive further apart than the specified value. The value is Global, or ranging from 50 to 30000 ms. First tone timeout: If selected, DTMF detection is terminated when the first tone arrives later than the specified amount of time. The value is Global, or ranging from 50 to 10000 ms. <p>Threshold Value: If selected, this parameter configures a threshold value for DTMF detection, whereby DTMFs having the duration smaller than the threshold value are not detected. The value range is 50 to 10000 ms.</p>
----------------------	--

Detect DTMF Properties: Output Settings

The following table describes the Detect DTMF Output Settings parameters:

Properties	Description
Connected	The DTMF detection was correctly executed for the specified number of times or duration. The default resolution for this output is SUCCESS.
Timeout	The DTMF detection failed because of a timeout (for example, DTMF/MF/Tone detection timeout). The default resolution for this output is WARNING.
Error	The function has returned an error. The default resolution for this output is FAILED.

Generate MF

This script function generates a sequence of Multi - Frequency (MF) tones. The sequence can have up to 31 tones and the tone length and the inter-tone Interval can be specified. The sequence can contain any combination of the standard tones, (that is, '1,' '2,' '3,' '4,' '5,' '6,' '7,' '8,' '9,' '0,' '#,' '*', 'a,' 'b,' 'c'.')

Because its parameters are similar to these of the Generate DTMF script function, see [Generate DTMF](#) for detailed information.

Detect MF

This script function is used to detect a sequence of MF tones. The sequence can have up to 31 tones and the inter-tone interval can be specified. Note that the inter-tone interval includes the silence time between two consecutive tones. The expected sequence to be received can contain any of the standard tones (that is, '1,' '2,' '3,' '4,' '5,' '6,' '7,' '8,' '9,' '0,' '#,' '*', 'a,' 'b,' 'c'.')

Because its parameters are similar to these of the Detect DTMF script function, see Detect DTMF for detailed information.

Generate Tone

This script function generates a single custom tone (single or dual continuous, or cadence) that can be selected from the Custom Tones Pool. The tones frequency must be between 300 and 3500 Hz, and, for dual tones, the difference of frequency between the two tones must be at least 65 Hz.

Generate Tone Properties: Parameters

The following table describes the Generate Tone function parameters:

Name	Description
Delay Before Execution	Delays the function execution by a value specified as either of the following: <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 2000, in milliseconds (ms)
Tone Name	Specifies a custom tone.
Tone duration	The time length of a DTMF tone in the tone sequence. The value is the global value configured in the Global Settings > T1/E1 > Voice window (for the Global option), or ranging from 50 to 10000 ms. The default value is 200 ms.

Generate Tone Properties: Advanced Settings

The following table describes the Generate Tone Advanced Settings parameters:

Name	Description
Playback	Specifies the tone playback mode using either of the options below, or by applying the values specified in the Global Settings > T1/E1 > Voice window (for the Use Global Setting option enabled): <ul style="list-style-type: none"> • Play: Specifies the number of times to play. • Repeat Continuously for: Specify the period of time to play. The default value is 1000 seconds. • Use Talk Time (All objectives except Channels): Plays the DTMF sequence for the duration of the Talk Time call parameter.

GenerateTone Properties: Output Settings

The following table describes the Generate Tone function Output Settings parameters:

Properties	Description
Output Settings	



Ok	The function was executed correctly executed for the specified number of times or duration. The default resolution for this output is SUCCESS.
Error	The function has returned an error. The default resolution for this output is FAILED.

Wait for Tone

This script function detects any custom tone from a user-configured tone list. For each expected tone that is configured in the list, a corresponding output is added to the script function; when a particular tone from the list is detected, the function exits on the corresponding output.

Wait for Tone Properties: Options

The following table describes the Wait for Tone function parameters:

Name	Description
Delay Before Execution	<p>Delays the function execution by a value specified as:</p> <ul style="list-style-type: none"> • Static Expression, in milliseconds (ms) • Random Between Expressions, in milliseconds (ms) • Sleep 1000, in milliseconds (ms) • GetCallInfo Delay, in milliseconds (ms) • Detect DTMF delay, in milliseconds (ms) • Generate DTMF delay, in milliseconds (ms) • Sleep 2000, in milliseconds (ms)
Tone Detection Settings	<p>This area contains a list of tones to be detected. Initially, the list is empty, adding a tone is done by clicking , which shows a dialog box that allows you to specify the expected tone. A tone can be deleted from the tone list by selecting its entry and clicking .</p> <p>Tone timeout: Specifies a timeout by entering a custom value, or by applying the global timeout settings configured in the Global Settings > T1/E1 > Voice window (for the Global value).</p>

Wait for Tone Properties: Output Settings

The following table describes the Wait for Tone Output Settings parameters. As stated before, a new script function output with the same name as the expected custom tone is added.

Properties	Description
Output Settings	
Custom Tone #1	The specified custom tone (Custom Tone #1) was detected. The default resolution for this output is SUCCESS.

Custom Tone #2	The specified custom tone (Custom Tone #2) was detected. The default resolution for this output is SUCCESS.
Timeout	The detection operation failed due to a timeout. The default resolution for this output is WARNING.
Error	The function has returned an error. The default resolution for this output is FAILED.

This page intentionally left blank.

CHAPTER 4 Basic Test Scenarios and Procedures

This chapter describes the built-in procedures and sample test configurations provided with the IxLoad Voice Plug-in and covers the following topics:

- SIP Procedures, Sample Test Configurations and Test Scenarios
- Skinny Procedures, Sample Test Configurations and Test Scenarios
- H.323 Sample Test Configurations and Test Scenarios
- H.248 Sample Test Configurations and Test Scenarios
- MGCP Sample Test Configurations and Test Scenarios
- PSTN Sample Test Configurations and Test Scenarios

SIP Procedures, Sample Test Configurations, and Test Scenarios

Using the functions from the VoIP SIP test library, you can generate and execute a large number of originator/answerer scenario configurations that comply with the SIP protocol. You can use one of the predefined test scenarios described in this chapter or create a new one, map it to VoIPSIP Peer activities, and start the execution.

This section describes the pre-defined IxLoad Voice Plug-in SIP procedures, available sample test configurations (RXFs) and their associated test scenarios.


Note: For a complete description of the supported SIP test library functions, see [VoIP SIP Functions Library](#).

Note: Sample tests follow a naming convention that comprises test type (VS for VoIP SIP and SC for SIP Cloud), an index, a test topology, a configuration (B2B for tests in back-to-back mode or DUT for tests running against a DUT), a protocol version (SIPv4 or SIPv6) and a short call description, for example, SC_001_B2B_SIPv4_T1_MakeCall_from_Cloud.


SIP Predefined Procedures

A procedure is a simple way to encapsulate several script functions into a single function block that can be reused within a number of scenarios. Based on the functions from the SIP test library, the SIP predefined procedures described in the following table (available in the Procedure Library) were developed:

Category	Procedure	Description
Registration	SIP MakeRegistration	A registration procedure that resolves 1xx and 200 response messages.
	SIP Make Registration - Authentication	A registration procedure resolving 1xx, 200, and 401 response messages. The procedure is used in test scenarios that require authentication.
	SIP Make Registration - 407 - Authentication	A registration procedure similar to the previous one, except that it additionally resolves 407 response messages.
	SIP MakeRegistration - First Loop Only	A registration procedure similar to the previous MakeRegistration - Authentication procedure, except that the SIP endpoint registers only during the first loop execution.

	SIP MakeRegistration 407 - First Loop Only	A registration procedure similar to the previous one, executing registration during the first loop only. The procedure additionally resolves 407 response messages.
	SIP Make Registration - Complete	A registration procedure that performs registration for the first loop only and resolves 1xx, 200, 401 and 5xx responses. When retransmissions are required, they need to be enabled at activity level.
	SIP Make ReRegistration - Authentication	A registration procedure that performs an initial registration during the first loop and retrieves the value of the expires SIP parameter or header. Starting with the second loop, the procedure evaluates the expiration timer and performs re-registration if the current registration is to expire during the current loop.
	SIP Unregister	A de-registration procedure that unregisters only one binding associated with an address-of-record.
	SIP Unregister All Bindings	A de-registration procedure that removes all the previous bindings currently in place for an address-of-record.
Originate Call	SIP MakeCall	A call originating procedure that comprises the Send INVITE, Wait 1xx, Wait 200, Send ACK functions sequence.
	SIP Make Call - Authentication	A call originating procedure that comprises the Send INVITE, Wait 1xx, Wait 200, or Wait 407, Send ACK, Resend INVITE with Authentication, Wait 1xx or Wait 200, Send ACK functions sequence.
	SIP Make Call - Route	A call originating procedure that uses routing information. This is the same as the previous Make Call - Authentication procedure, except that the Send ACK script function includes the SIP Route message header. This procedure is used in test scenarios that require routing functionality.
	SIP Make Call - Redirect Server	A call originating procedure that resolves 3xx response messages and uses redirect information. This is the same as the previous Make Call - Authentication procedure, except that it is used in scenarios that involve a redirect server.
	SIP Make Call - Complete	A call originating procedure that resolves a wide range of responses: 1xx, 200, 3xx, 4xx, 5xx, and 6xx.  Note: To use the message retransmission mechanism, this procedure needs to have retransmissions enabled at activity level.

Receive Call	SIP Receive Call	A call receiving procedure that comprises the Wait INVITE, Send 180 Ringing, Send 200, Wait ACK functions sequence.
	SIP Receive Call - Busy Here	A call receiving procedure that sends a 486 Busy here message in response to a SIP INVITE message.
	SIP Receive Call - No Answer	A call receiving procedure that sends a 180 Ringing message in response to a SIP INVITE message and then waits for a SIP CANCEL message.
	SIP Receive Call - Record Route	A call receiving procedure similar to the previous Receive Call procedure, except that SIP response messages include the Record-Route message header field. This procedure is used in test scenarios that require routing functionality.
End Call	SIP End Call Initiate	A simple call disconnection procedure that comprises a Send BYE and Wait 200 functions sequence.
	SIP End Call Initiate - Route	A call disconnection procedure similar to the previous End Call Initiate procedure, except that it uses routing information. This procedure is used in test scenarios that require routing functionality.
	SIP End Call Receive	A call disconnection procedure for the receiving side, which comprises a Wait BYE and Send 200 OK functions sequence.
	SIP End Call Receive - Record Route	A call disconnection procedure similar to the previous procedure, except that the Send 200 OK function includes a Record-Route header. This procedure is used in scenarios that require routing functionality.
Hold/Unhold	SIP Hold - Initiate	A procedure that puts the remote party on hold and comprises a Send INVITE (Hold Session message body), Wait 200 OK, Send ACK functions sequence.
	SIP UnHold - Initiate	A procedure that unholds the remote party and comprises a Send INVITE, Wait 200 OK, Send ACK commands sequence.
	SIP Hold - Receive	A procedure that puts the party on hold. It comprises a Wait INVITE, Send 200 OK, Wait ACK commands sequence.
	SIP UnHold - Receive	A procedure that takes the party off hold. The procedure comprises a Wait INVITE, Send 200 OK, Wait ACK commands sequence.

IMS	SIP IMS MakeRegistration	<p>An IMS-compliant registration procedure that uses custom message headers and user-defined scenario variables.</p> <hr/> <p> Note: The scenario variables used by the procedure are described in the procedure body.</p> <hr/>
	SIP IMS MakeCall	An IMS calling procedure for the originating side that comprises the Send INVITE, Wait 100 Trying or 183 Session in Progress, Send PRACK, Wait 200 OK for PRACK, Send UPDATE, Wait 200 OK for UPDATE, Wait 180 Ringing, Send PRACK, Wait 200 OK for PRACK, Wait 200 OK for INVITE, Send ACK functions sequence.
	SIP IMS ReceiveCall	An IMS procedure for the receiving side, comprising the Wait INVITE, Send 100 Trying or 183 Session in Progress, Wait PRACK, Send 200 OK or PRACK, Wait UPDATE, Send 200 OK for UPDATE, Send 180 Ringing, Wait PRACK, Send 200 OK for PRACK, Send 200 OK for INVITE, Wait ACK messages sequence.
	SIP IMS EndCall Initiate	An IMS call terminating procedure for the originating side comprising a Send BYE, Wait 200 OK functions sequence.
	SIP IMS EndCall Receive	An IMS call terminating procedure for the receiving side comprising a Wait BYE, Send 200 OK functions sequence.
	SIP IMS Subscribe	An IMS subscription procedure that comprises the Send SUBSCRIBE, Wait 200 OK, Wait NOTIFY, Send 200 OK for NOTIFY (Wait multiple NOTIFY) functions sequence.
Instant Messaging Support	SIP Send Instant Message	An instant message sending procedure that comprises the Send MESSAGE and Wait 200 OK functions sequence.
	SIP Wait Instant Message	An instant message receiving procedure that comprises the Wait MESSAGE and Send 200 OK functions sequence.
SMS	SIP SMS Deliver Initiate	SIP SMS Deliver Initiate is used to simulate CSCF behavior for sending Deliver messages. The procedure comprises Send Deliver, Wait 200OK, Wait Deliver-Report, and Send 202 Accepted functions set.
	SIP SMS Deliver Receive	SIP SMS Deliver Receive is used to simulate Endpoint behavior for receiving Deliver messages. The procedure comprises Wait Deliver, Send 200 OK, Send Deliver-Report and Wait 202 Accepted functions set.

	SIP SMS Status-Report Initiate	SIP SMS Status-Report Receive is used to simulate CSCF behavior for sending Status-Report messages. The procedure comprises Send Status- Report, and Wait 200 OK functions set.
	SIP SMS Status-Report Receive	SIP SMS Status-Report Receive is used to simulate Endpoint behavior for receiving Status-Report messages. The procedure comprises Wait Status- Report, and Send 200 OK functions set.
	SIP SMS Submit Initiate	SIP SMS Submit Initiate is used to simulate Endpoint behavior for sending Submit messages. The procedure comprises Send Submit, Wait 202 Accepted, Wait Submit-Report, and Send 200 OK functions set.
	SIP SMS Submit Receive	SIP SMS Submit Receive is used to simulate CSCF behavior for receiving Submit messages The procedure comprises Wait Submit, Send 202 Accepted, Send Submit-Report and Wait 200 OK functions set.
Other	First Loop?	A procedure that tests if the current loop is the first loop.

VoIPSIP Peer Test Configurations

The following section provides you with a short description of the SIP test scenarios that are included in the IxLoad installation kit. This section provides a brief description of the sample VoIPSIPPeer test configuration files contained in the IxLoad installation kit.

Note: Sample SIP test configurations do not have retransmissions enabled at activity level. For complex tests that require retransmissions you can also enable retransmissions at function level.

Note: When configuring SIP UAs, keep in mind that a signaling endpoint has to be configured using a unique (IP address, Port, Phone number) tuple, while a media endpoint is uniquely identified by a (IP Address, Port) tuple.

Note: Sample tests provided with the IxLoad Voice Plug-in normally do not resolve 407 response messages. In cases when your tests require the resolving of 407 messages, predefined registration procedures from the sample scenarios can be replaced with the SIP Make Registration - 407 - Authentication and the SIP Make Registration - 407 - First Loop Only procedures that handle such messages.

VS_001_B2B_SIPv4 MakeCall - ReceiveCall - EndCall

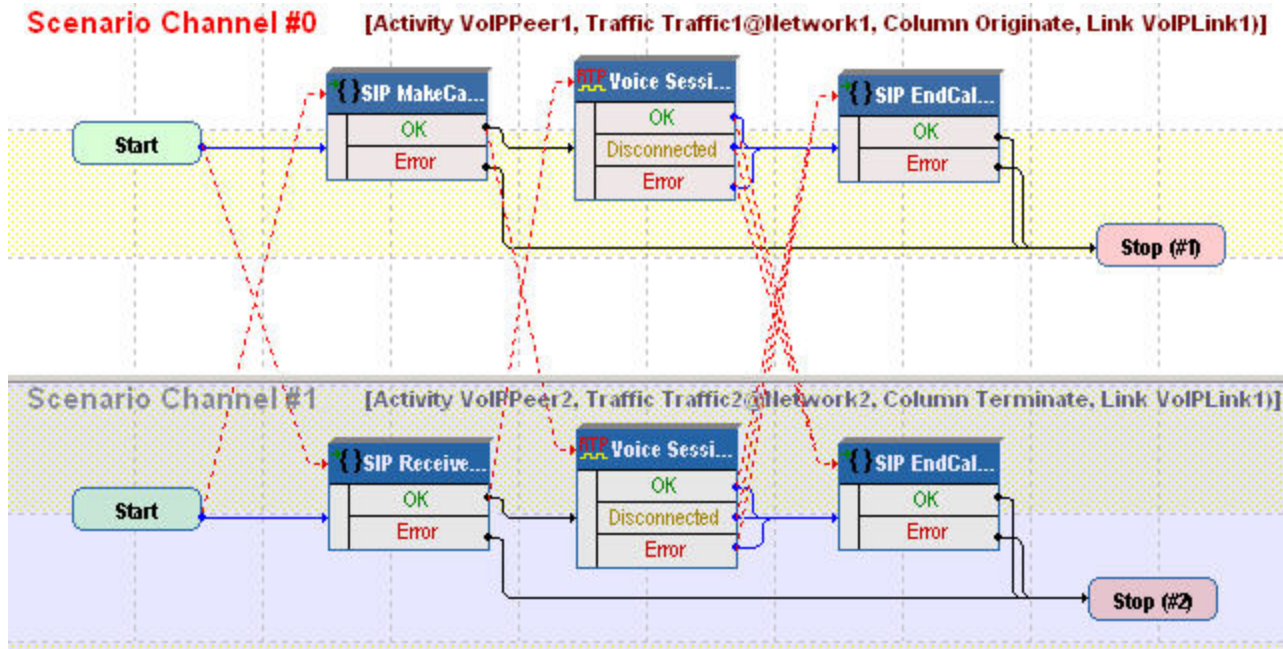
This test is similar to the following [VS_002_B2B_SIPv4 MakeCall - ReceiveCall - EndCall with RTP - 33s](#) test, except that in the underlying test scenario Sleep script function are used instead of the Voice Session functions.

Note: The SIP sample templates include a VS_034_B2B_SIPv6 MakeCall - ReceiveCall - EndCall predefined test identical to this one, except that it uses IPv6 network settings.

VS_002_B2B_SIPv4 MakeCall - ReceiveCall - EndCall with RTP - 33s

This test runs in Back-to-Back mode and comprises two VoIPSIPPeer activities, Make_Call and Receive_Call.


Make_Call is linked to a test scenario channel that establishes a SIP call with media streaming by using Voice Session script functions and then disconnects, as shown in the following image. Receive_Call executes the test flow on the receiving side.




The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario	<p>The VS_002_B2B_SIPv4 MakeCall - ReceiveCall - EndCall with RTP - 33s test scenario comprises two channels executing a basic call procedure with media exchange.</p> <p>Channel#0: MakeCall, Voice Session, EndCall Initiate.</p> <p>Channel#1: ReceiveCall, Voice Session, EndCall Terminate.</p>
Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) <p>For each media channel, a unique (IP, port) tuple is generated using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value


Dial Plan	The Receive_Call activity and port 5060 are configured as call destination.
SIP Settings	The SIP port has the 5060 default setting for all channels.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is selected for media streaming to be performed. Because Make_Call emulates media endpoints by using different IP addresses, a single RTP port (10000) is configured for media streaming.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** Receive_Call uses the same settings as Make_Call, except for the Dial Plan page, which does not need to specify a destination activity, because it only terminates a call.

 **Note:** The SIP sample templates include a VS_035_B2B_SIPv6 MakeCall - ReceiveCall - EndCall with RTP - 33s predefined test identical to this one, except that it uses IPv6 network settings.

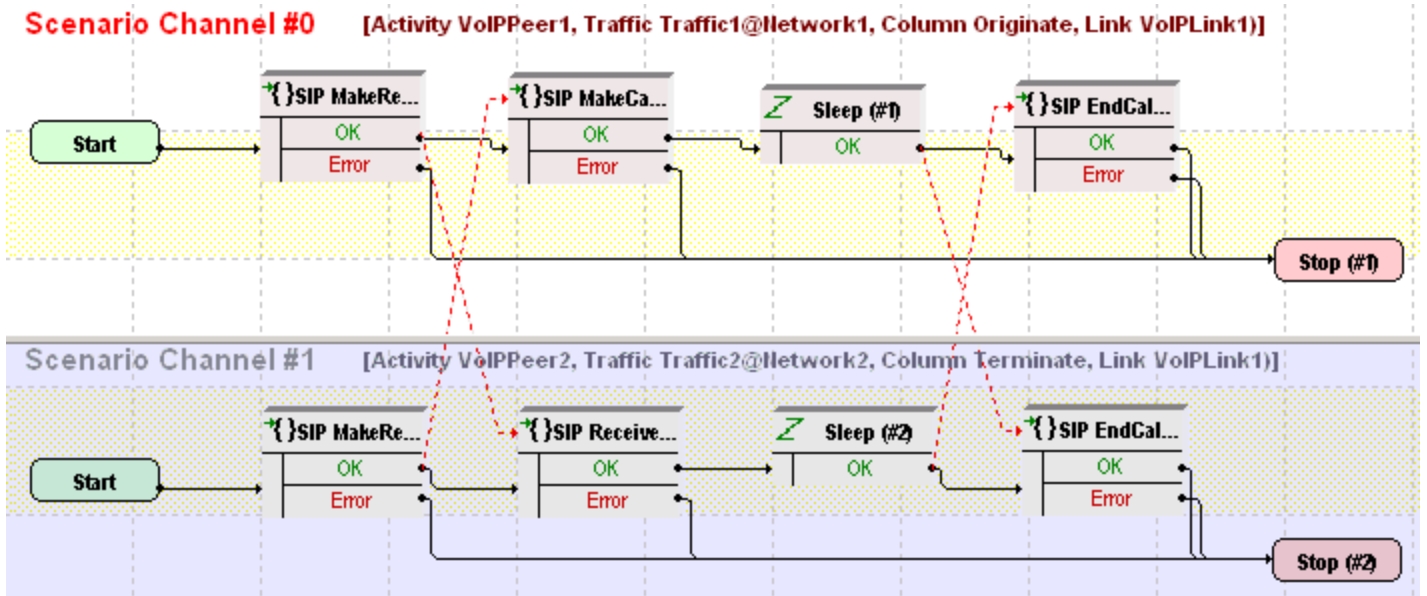
VS_003_B2B_SIPv4 MakeCall - ReceiveCall - EndCall with SRTP - 33s

This test is similar to the previous [VS_002_B2B_SIPv4 MakeCall - ReceiveCall - EndCall with RTP - 33s](#) test, except that the media traffic is encrypted by using SRTP.

 **Note:** The SIP sample templates include a VS_036_B2B_SIPv6 MakeCall - ReceiveCall - EndCall with SRTP - 33s predefined test identical to this one, except that it uses IPv6 network settings.

VS_004_B2B_SIPv4 MakeCall - ReceiveCall - EndCall - Early Media

This test is similar to the previous [VS_002_B2B_SIPv4 MakeCall - ReceiveCall - EndCall with RTP - 33s](#) test, except that the media capabilities are negotiated on a provisional 183 SessionInProgress response instead of a 200 OK response.



The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario	The VS_005_DUT_SIPv4 MakeCall - ReceiveCall - EndCall with RecordRoute test scenario is completely configured. The SIP MakeCall - Route and SIP EndCall Initiate - Route procedures used on the first channel are used for configuring message routing functionality.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The Receive_Call activity and port 5060 are configured as call destination.
SIP Settings	The SIP port has the 5060 default setting for all channels. The SIP server the test runs against is configured in the Use external server area.
Codec Settings	The default codec settings can be left unaltered.
RTP Settings	Because this test does not perform any media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

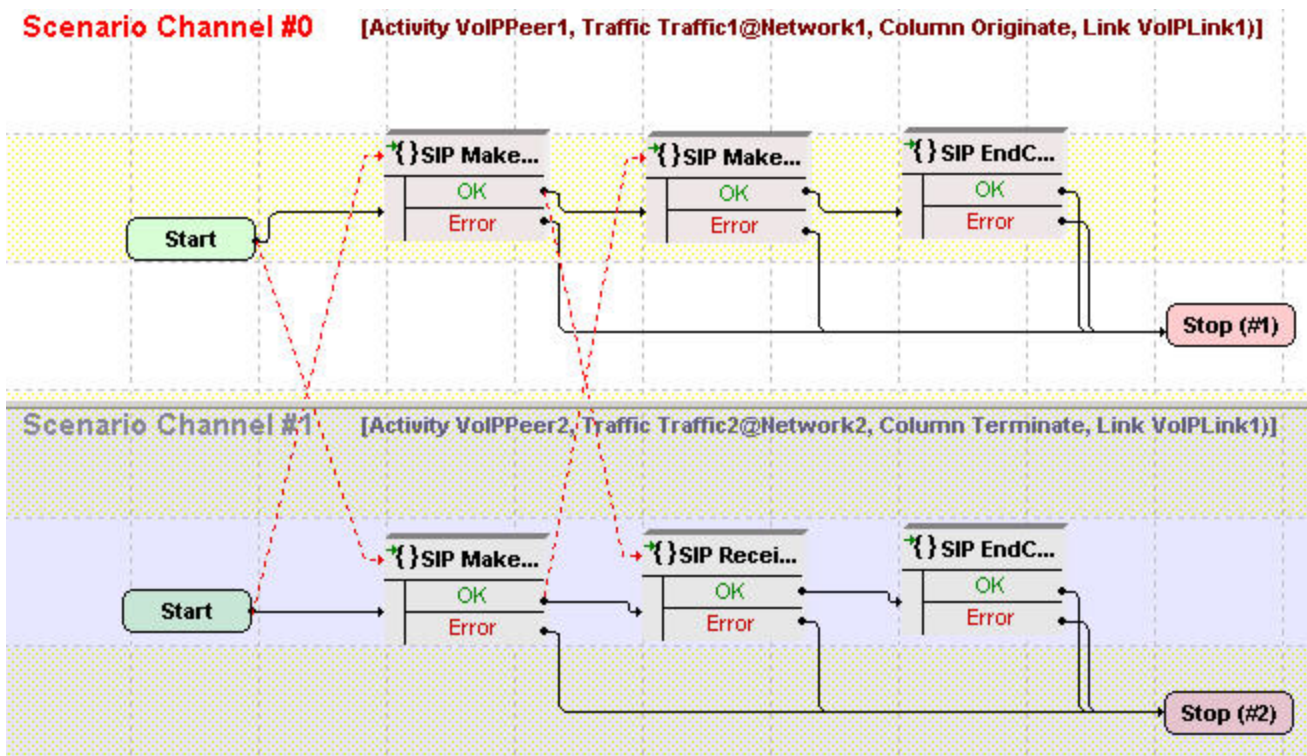
Note: Receive_Call uses the same settings as Make_Call, except for the Dial Plan page, which does not need to specify a destination activity, because it only terminates a call.

Note: The SIP sample templates include another VS_039_DUT_SIPv6 MakeCall - ReceiveCall - EndCall with RecordRoute predefined test identical to this one, except that it uses IPv6 network settings.

VS_006_DUT_SIPv4 MakeCall - ReceiveCall with Registration


This test runs against a SIP Server and comprises two VoIPSIPPeer activities, Make_Call and Receive_Call.


Make_Call is linked to a test scenario channel that registers with a SIP Proxy server, establishes a SIP call without media traffic, and disconnects, as shown in the following image. Receive_Call executes the test flow for the receiving side.




Make_Call configuration settings are described in the following table:

Category	Settings
Scenario	<p>The VS_006_DUT_SIPv4 Make - Receive Call with Registration test scenario is completely configured and supports authentication at test scenario level. The scenario comprises two channels:</p> <p>Channel#0: Make Registration, MakeCall - Authentication, EndCall Initiate.</p> <p>Channel#1: Make Registration, ReceiveCall, EndCall Receive.</p>

Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The Receive_Call activity and port 5060 are configured as call destination.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port = Use same value setting from the Execution Settings page. The SIP server you are running this test against must be configured in the Use external Server area. <hr/> <p> Note: Because the test scenario itself supports authentication, you can configure the test to use authentication and enter the desired authentication settings in the Authentication UAC area.</p> <hr/>
Codec Settings	The codec settings can be left unaltered.
RTP Settings	Because this test does not perform any RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

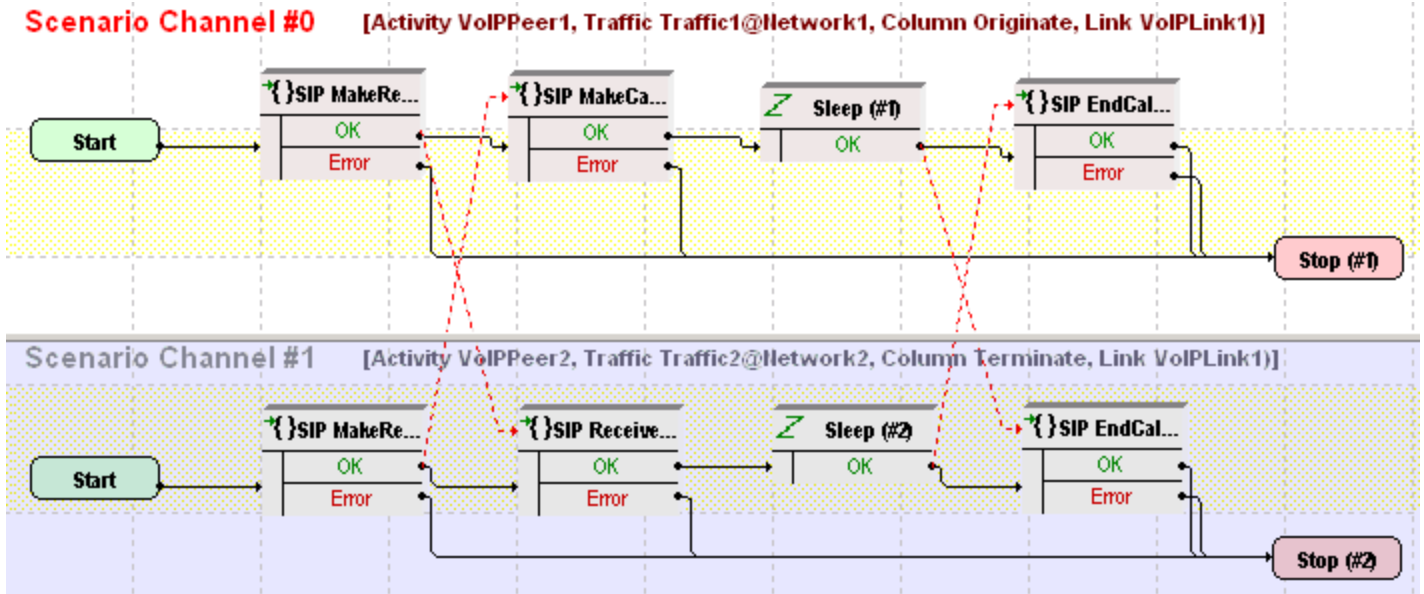
 **Note:** Receive_Call uses the same settings as Make_Call, except for the Dial Plan page, which does not need to specify a destination activity, since it only terminates a call.

 **Note:** The SIP sample templates include a VS_037_DUT_SIPv6 MakeCall - ReceiveCall with Registration predefined test identical to this one, except that it uses IPv6 network settings.

VS_007_DUT_SIPv4 Make - Receive Call with ReRegistration

This test runs against a SIP Server and comprises two VoIPSIPPeer activities, Make_Call and Receive_Call.

Make_Call emulates an endpoint that registers with a SIP server by using the SIP Make ReRegistration - Authentication procedure, establishes a signaling-only call with another endpoint emulated by Receive_Call, and disconnects, as shown in the following image. Receive_Call executes the test flow for the receiving side.



The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario	<p>The VS_007_DUT_SIPv4 Make - Receive Call with ReRegistration test scenario is completely configured and supports authentication at scenario level. The scenario comprises two channels.</p> <p>Channel#0: Make Registration, MakeCall - Authentication, EndCall Initiate.</p> <p>Channel#1: Make Registration, ReceiveCall, EndCall Receive.</p>
Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	<p>The Receive_Call activity and port 5060 are configured as call destination.</p>
SIP Settings	<p>The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port = Use same value setting from the Execution Settings page. The SIP server you are running this test against must be configured in the Use external Server area.</p> <p>Note: Because the test scenario itself supports authentication, you can configure the test to use authentication and enter the desired authentication settings in the UAC Authentication area.</p>

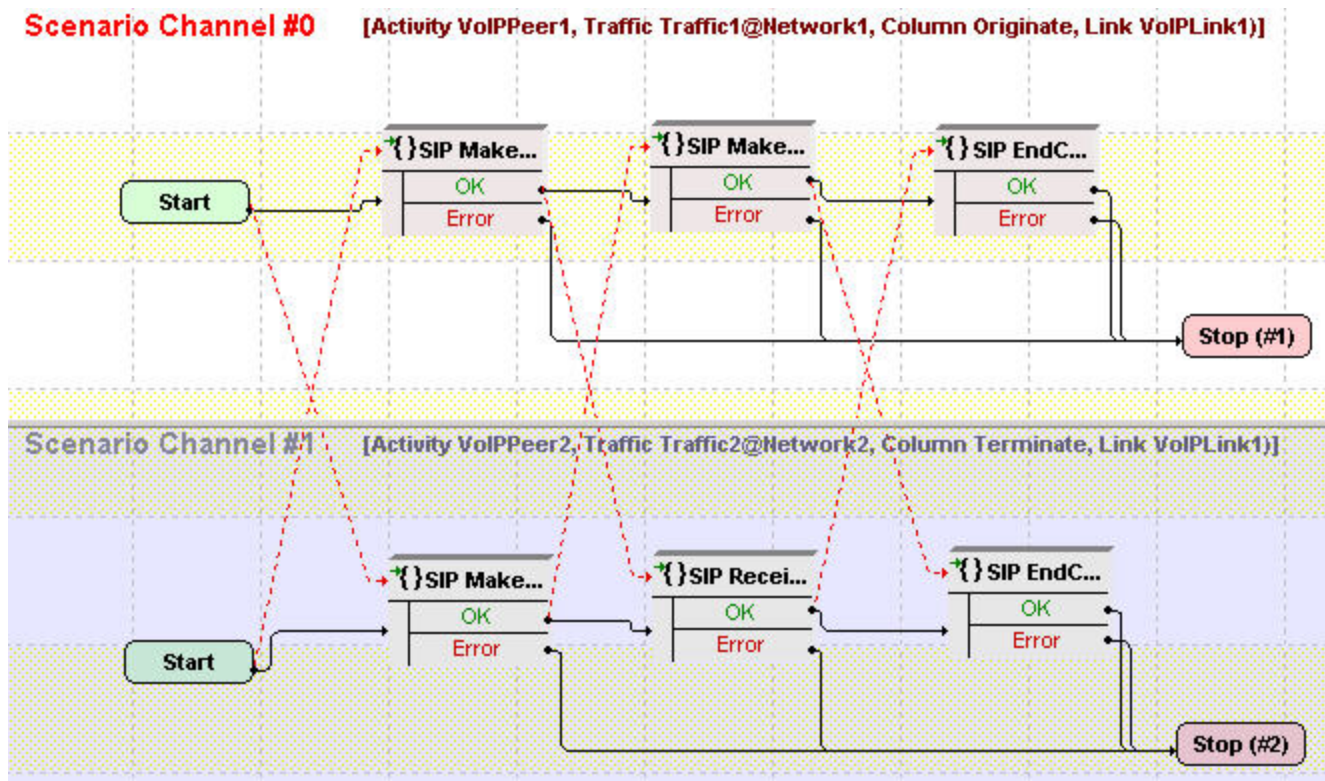
Codec Settings	The codec settings can be left unaltered.
RTP Settings	Because this test does not perform any RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Receive_Call uses the same settings as Make_Call, except for the Dial Plan page, which does not need to specify a destination activity, because it only terminates a call.

VS_008_DUT_SIPv4 MakeCall - Receive Call with Registration - Complete

This test runs against a SIP Server and comprises two VoIPSIPPeer activities, Make_Call and Receive_Call.

Make_Call is linked to a test scenario channel that emulates an endpoint that registers with a SIP server, establishes a SIP call with an endpoint emulated by Receive_Call, and then disconnects, as shown in the following image. Receive_Call executes the test flow on the receiving side.



Note: The SIP sample templates include a VS_038_DUT_SIPv6 MakeCall - ReceiveCall with Registration - Complete predefined test identical to this one, except that it uses IPv6 network settings.

VS_009_B2B_SIPv4 MakeCall - ReceiveCall with Tel URI - Global Phone Numbers

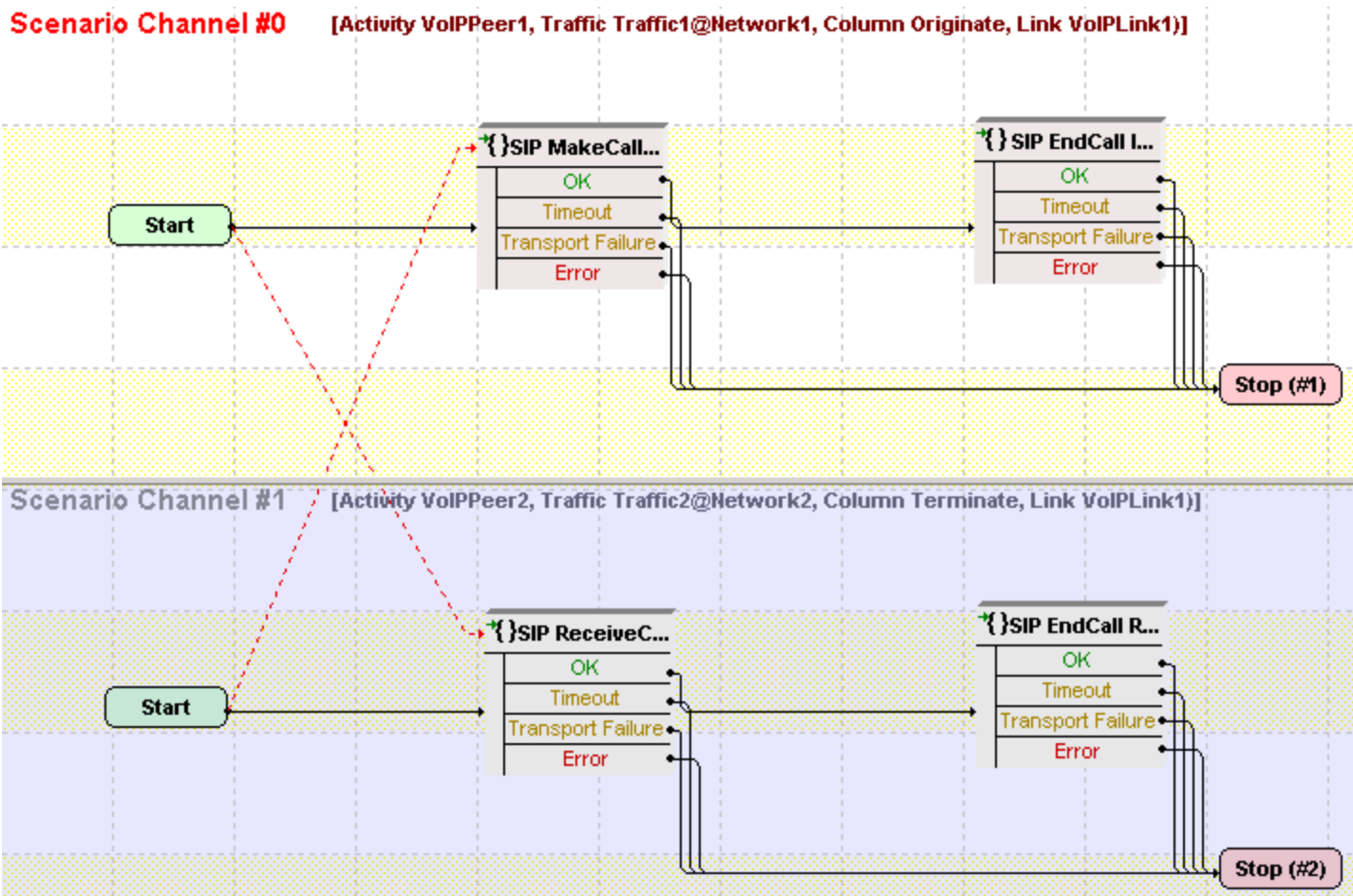
This test is identical to the following [VS_010_B2B_SIPv4 Make - Receive Call with Tel URI - Local Phone Numbers](#) test, except that it uses a global tel URI destination in the Dial Plan page. When a global tel URI is used, no additional parameters need to be specified.

Note: The underlying test scenario is the same for both this and the following [VS_010_B2B_SIPv4 Make - Receive Call with Tel URI - Local Phone Numbers](#) test configuration.

VS_010_B2B_SIPv4 Make - Receive Call with Tel URI - Local Phone Numbers

This test runs in Back-to-Back mode and comprises two VoIPSIPPeer activities, Make_Call and Receive_Call.

Make_Call is linked to a test scenario channel that establishes a call by using the Make Call - Authentication procedure, and disconnects by using the End Call Initiate procedure, as shown in the following image. Receive_Call executes the corresponding operations on the receiving side.



The Make_Call settings are described in the following table:

Category	Settings
----------	----------

Scenario Editor	The VS_010_B2B_SIPv4 MakeCall - ReceiveCall with Tel URI test scenario is completely configured, no further configuration is necessary.
Execution Settings	The channels are configured to use consecutive IPs from those available in the Network settings. For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The Receive_Call activity is configured as call destination. Because the test uses local Tel URIs, the phone-context parameter according to RFC 3966 must be defined for the phone number sources.
SIP Settings	The SIP port has the 5060 default setting. In the Construction of SIP messages area, the Use Tel URI scheme for source and Use Tel URI scheme for destination options are selected.
Codec Settings	The codec settings can be left unaltered.
RTP Settings	Because this test does not perform any RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4.

The Receive_Call settings are described in the following table:

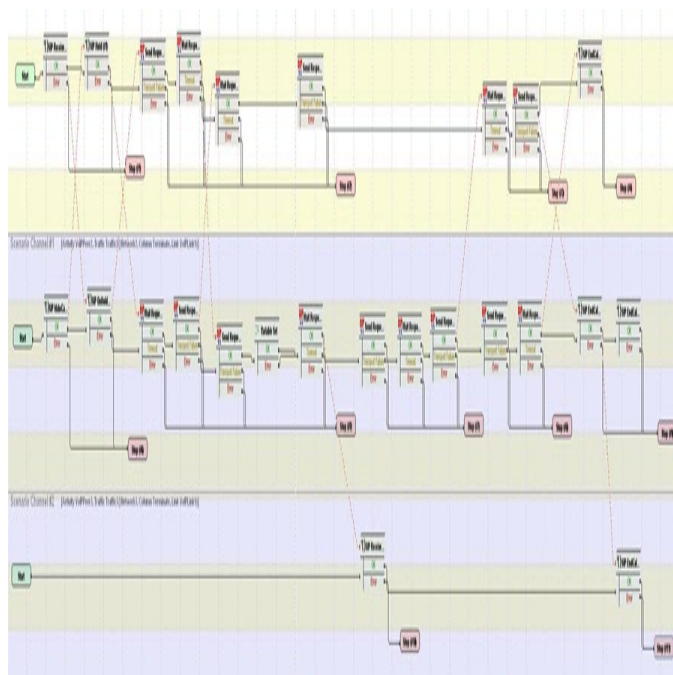
Category	Settings
Scenario Editor	The VS_010_B2B_SIPv4 MakeCall - ReceiveCall with Tel URI test scenario is completely configured, no further configuration is necessary.
Execution Settings	The channels are configured to use consecutive IPs from those available in the Network settings. For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	For this activity which only terminates a call, no VoIPSIPPeer activity is configured as destination activity. Source phone numbers are defined as a sequence generator expression. Because the test uses local Tel URIs, the phone-context parameter according to RFC 3966 must be defined for the phone number sources.

SIP Settings	The SIP port has the 5060 default setting. In the Construction of SIP messages area, Use Tel URI scheme for source is selected.
Codec Settings	The codec settings can be left unaltered.
RTP Settings	Because this test does not perform any RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4.

VS_011_B2B_SIPv4 Basic Transfer - Successful

This test comprises three VoIPSIPPeer activities, VoIPSIPPeer1, VoIPSIPPeer2, and VoIPSIPPeer3.

VoIPSIPPeer2 (transferee) is linked to a test scenario channel that establishes a SIP call with VoIPSIPPeer1 (transferrer), and then transfers the call to VoIPSIPPeer3 (transfer target), as shown in the following image:



The VoIPSIPPeer1 configuration settings are described in the following table:

Category	Settings
----------	----------

Scenario	<p>The VS_011_B2B_SIPv4 Basic Transfer - Successful test scenario comprises three channels that implement a call transfer:</p> <ul style="list-style-type: none"> • Channel#0: This is the receiving party for the call initiated by endpoint B. Endpoint A (the transferrer) puts the established call on hold and transfers it to C. • Channel#1: This is the originating party for a call to endpoint A. After the call is established, endpoint B (the transferee) is transferred to endpoint C. • Channel#2: Endpoint C is the transfer target.
Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	<p>Because the activity only terminates a call, no activity is configured as call destination.</p>
SIP Settings	<p>The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port = Use same value setting from the Execution Settings page.</p> <div style="border: 1px solid gray; padding: 5px; margin-top: 10px;"> <p>! Important! The address of the VoIPSIP Peer activity to which the call is transferred (VoIPSIPPeer3) is configured as Transfer Address in the SIP Settings page.</p> </div>
Codec Settings	<p>The codec settings can be left unaltered.</p>
RTP Settings	<p>Because this test does not perform any RTP streaming, the Enable media on this activity option is not selected.</p>
Other Settings	<p>The IP version preference is set to IPv4, and no scenario variables need to be initialized.</p>

Note: VoIPSIPPeer2 uses the same settings as VoIPSIPPeer1, except for the Dial Plan page, which specifies VoIPSIPPeer1 as call destination. The settings for VoIPSIPPeer3 are similar to those of VoIPSIPPeer1, except for the transfer address, which needs not be specified in the **SIP Settings** page.

Note: The SIP sample templates include another VS_041_B2B_SIPv6 Basic Transfer - Successful predefined test identical to this one, except that it uses IPv6 network settings.

VS_012_B2B_SIPv4 Basic Transfer - Target Busy

The test configuration level settings are the same as those for the previous VS_011_B2B_SIP v4 Basic Transfer - Successful test.

Note: The only difference between the VS_012_B2B_SIPv4 Basic Transfer - Target Busy and the VS_011_B2B_SIPv4 Basic Transfer - Successful tests is the underlying scenario, which simulates a transfer failure due to a busy target condition, instead of a successful transfer.

VS_013_B2B_SIPv4 Basic Transfer - Target No Answer

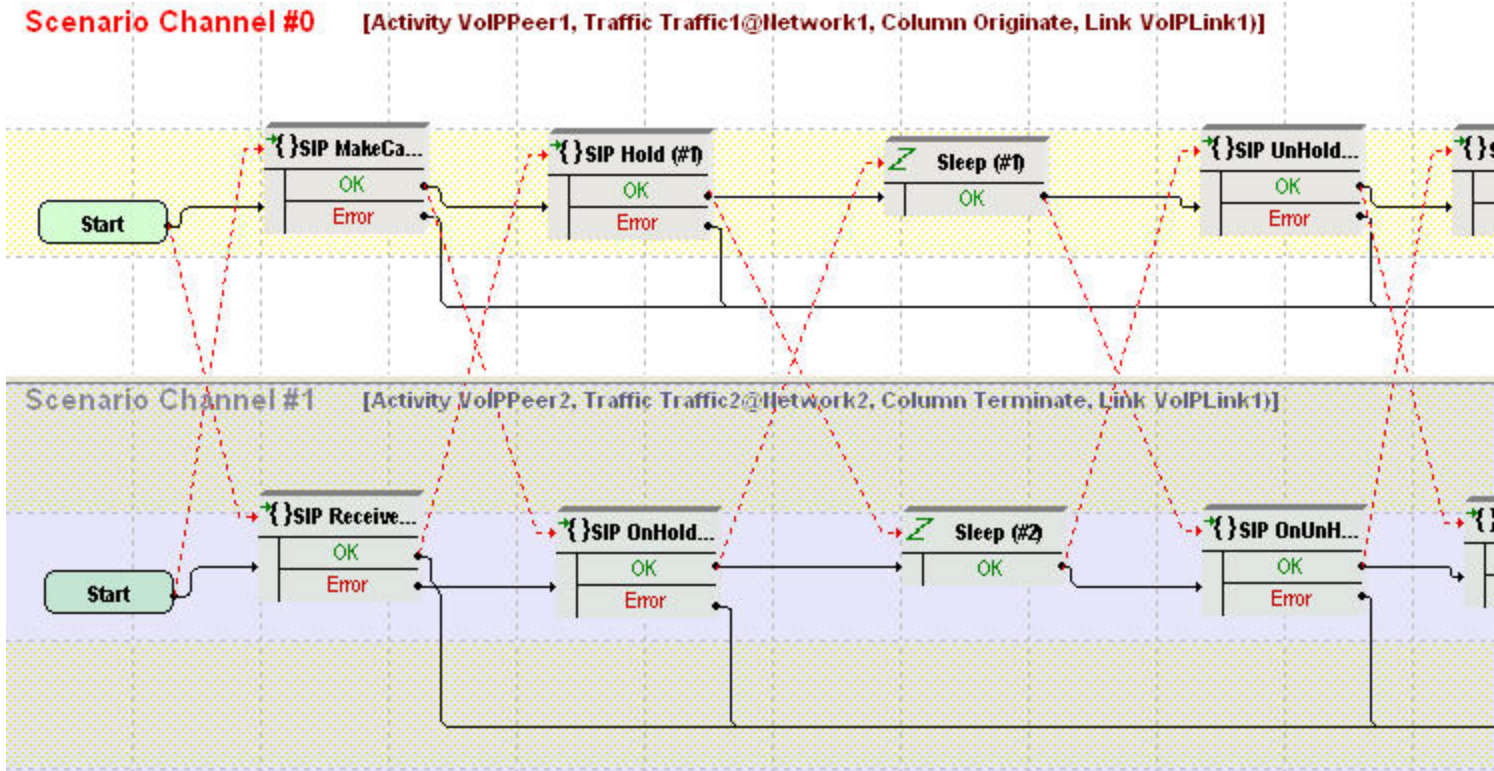
The test configuration level settings are the same as those for the previous VS_011_B2B_SIPv4 Basic Transfer - Successful test.

Note: The only difference between the VS_013_B2B_SIPv4 Basic Transfer - Target No Answer and the VS_011_B2B_SIPv4 Basic Transfer - Successful tests is the underlying scenario, which simulates a transfer failure due to a target no answer condition, instead of a successful transfer.

VS_014_B2B_SIPv4 MakeCall - ReceiveCall - EndCall with Hold UnHold

This test illustrating a Hold/Unhold procedure runs in Back-to-Back mode and comprises two VoIPSIPPeer activities, Make_Call and Receive_Call.

Make_Call is linked to a test scenario channel that establishes a SIP call without media streaming, performs a hold/unhold procedure by using the SIP Hold and SIP Unhold script functions, and then disconnects, as shown in the following image. Receive_Call executes the test flow on the receiving side.



The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario	The VS_014_B2B_SIPv4 MakeCall - ReceiveCall - EndCall with Hold - Unhold test scenario comprises two channels: Channel#0: MakeCall, SIP Hold - Initiate, Sleep, Unhold - Initiate, EndCall Initiate. Channel#1: ReceiveCall, Hold - Receive, Sleep, Unhold - Receive, EndCall Terminate.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The Receive_Call activity and port 5060 are configured as call destination.
SIP Settings	The SIP port has the 5060 default setting for all channels.
Codec Settings	The codec settings can be left unaltered.
RTP Settings	Because this test does not perform any RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

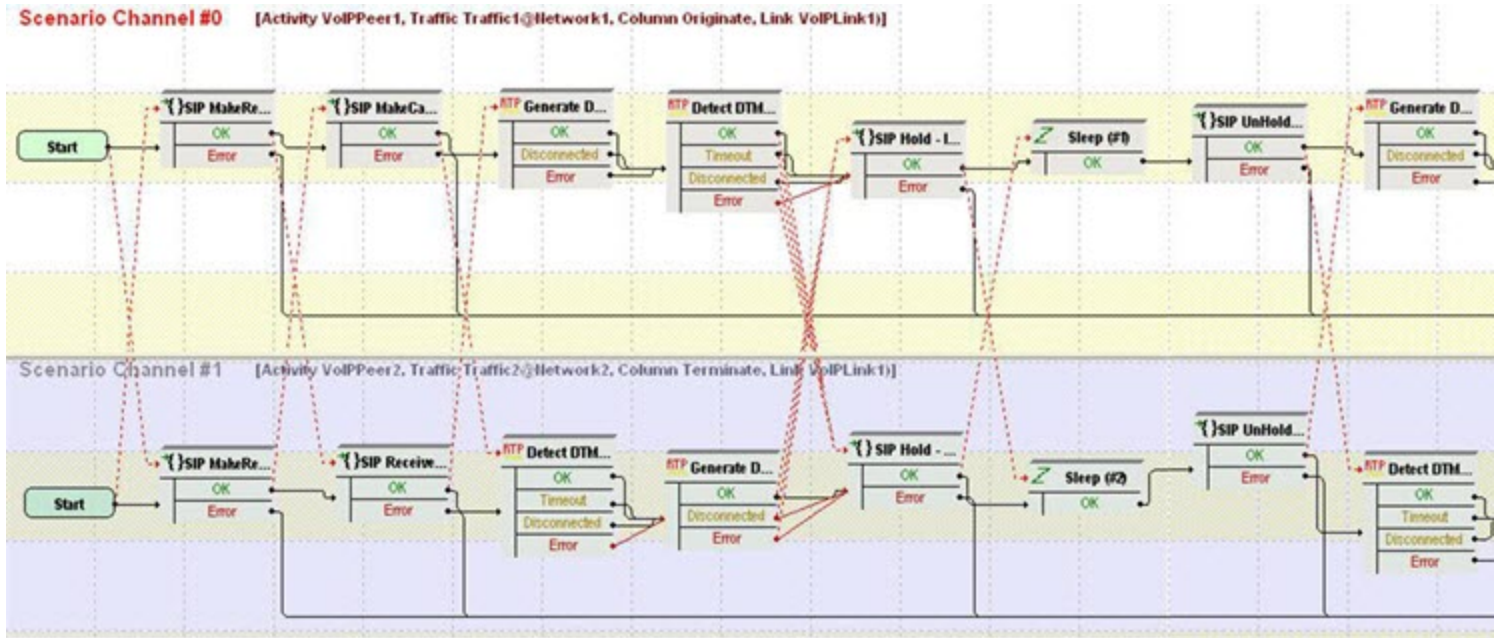
Note: Receive_Call uses the same settings as Make_Call, except for the Dial Plan page, which does not need to specify a destination activity, because it only terminates a call.

Note: The SIP sample templates include another VS_040_B2B_SIPv6 MakeCall - ReceiveCall - EndCall with Hold Unhold predefined test identical to this one, except that it uses IPv6 network settings.

VS_015_DUT_SIPv4 Hold - UnHold with Registration and Path Confirmation

This test runs against a SIP server and comprises two VoIPSIPPeer activities, Make_Call and Receive_Call.

Make_Call is linked to a test scenario channel that registers with a SIP server, establishes a call, performs audio path confirmation by using the Generate DTMF/Detect DTMF script functions pair, puts the remote party on hold, unholds the remote party, performs audio path confirmation again, and disconnects, as shown in the following image. Receive_Call executes the corresponding test flow on the receiving side.



The Make_Call configuration settings are described in the following table:

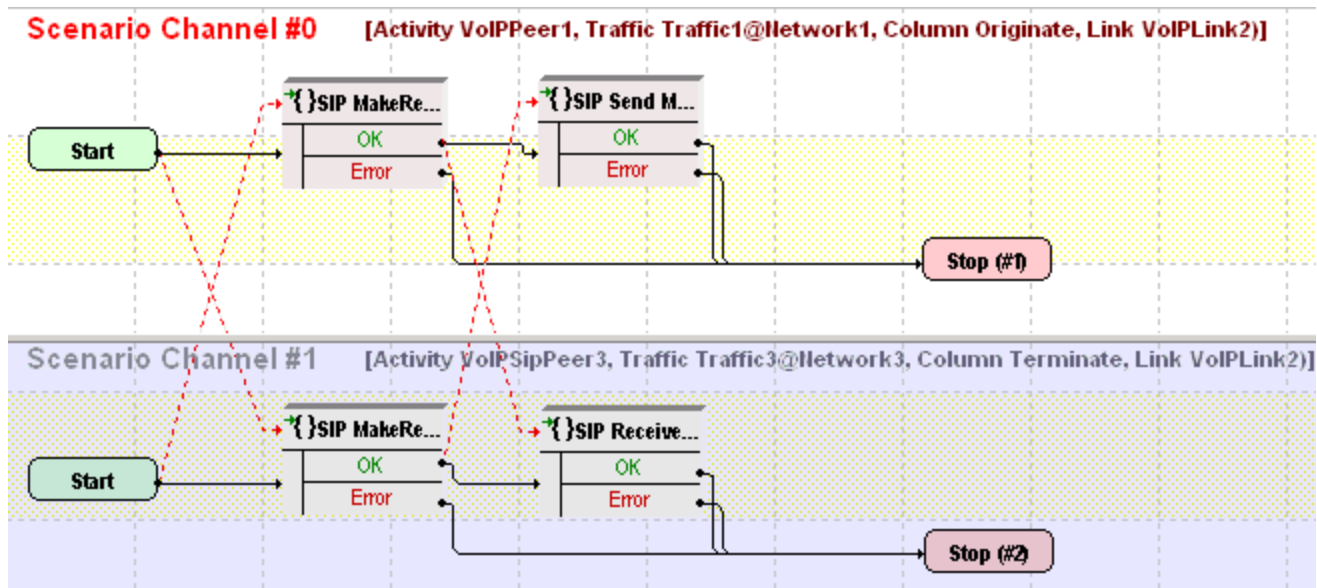
Category	Settings
Scenario	<p>The VS_015_DUT_SIPv4 Hold - UnHold with Registration, Path Confirmation test scenario is completely configured and supports authentication at test scenario level.</p> <p>Channel#0: Make Registration, Make Call, Generate DTMF, Detect DTMF, Hold, Sleep, UnHold, Generate DTMF, Detect DTMF, End Call Initiate.</p> <p>Channel#1: Make Registration, Receive Call, Detect DTMF, Generate DTMF, onHold, Sleep, onUnHold, Detect DTMF, Generate DTMF, End Call Receive.</p>
Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) <p>For each media channel, a unique (IP, port) tuple is generated using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • UDP port: Use same value
Dial Plan	The Receive_Call activity is configured as call destination.

SIP Settings	<p>The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port: Use same value setting from the Execution Settings page. The SIP server you are running this test against must be configured in the Use external server area.</p> <p>Note: Because the test scenario itself supports authentication, you can configure the test to use authentication and enter the desired authentication settings in the UAC Authentication area.</p>
Codec Settings	The codec settings can be left unaltered.
RTP Settings	Because this test uses RTP streaming on the Generate DTMF/Detect DTMF script functions, the Enable media on this activity option is selected. For this test, which uses consecutive values for the IP addresses, RTP ports for all channels can be configured to the default 10000 setting.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: Receive_Call uses the same settings as Make_Call, except for the Dial Plan page, which does not need to specify a destination activity, because it only terminates a call.


VS_016_DUT_SIPv4 Send - Receive MESSAGE with Registration


This test comprises two VoIPSIPPeer activities, Make_Call and Receive_Call, whose emulated endpoints register with a SIP Proxy server by using SIP MakeRegistration - First Loop Only procedures and then exchange a SIP MESSAGE message, as shown in the following image:



The Make_Call configuration settings are described in the following table:

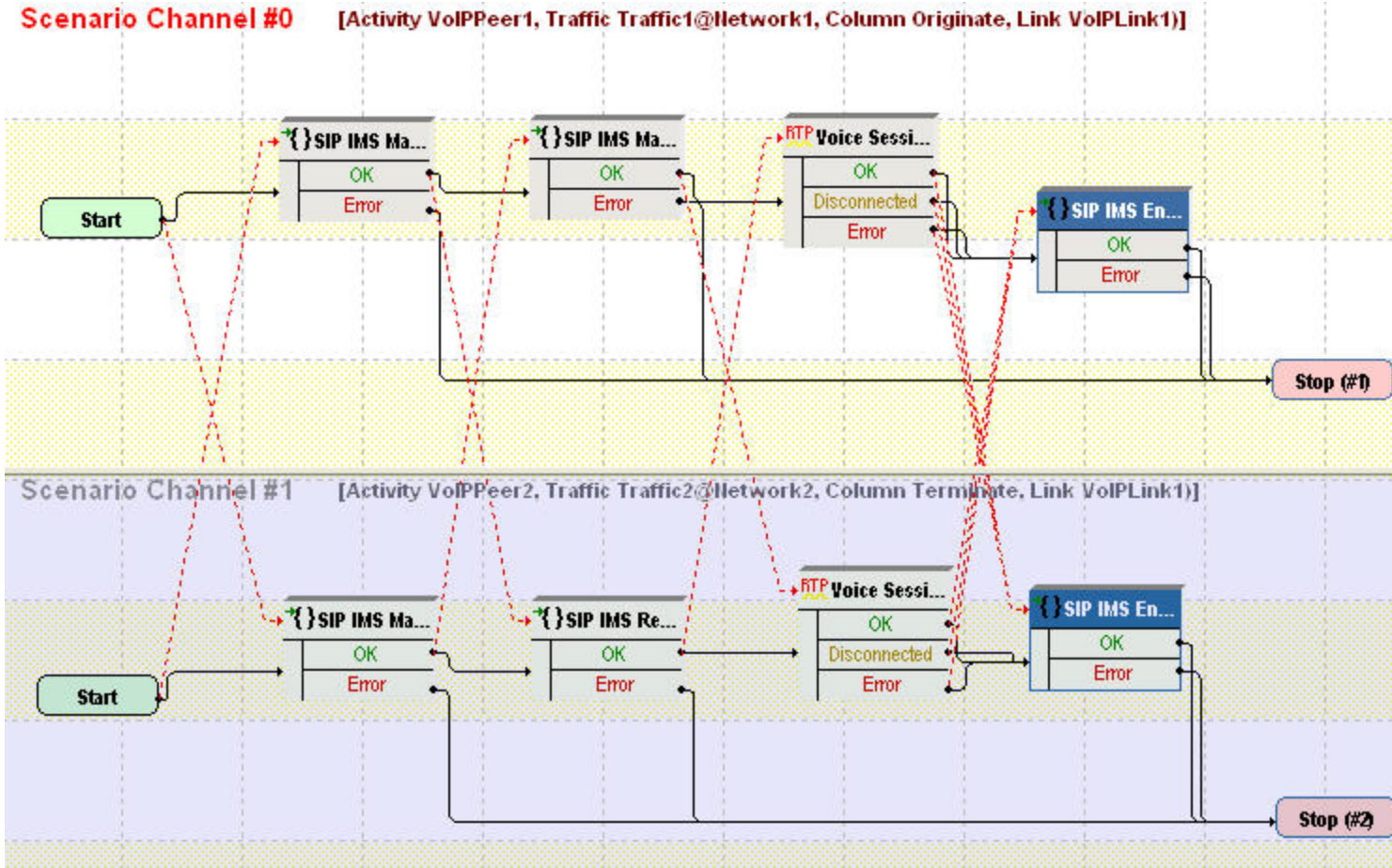
Category	Settings
Scenario Editor	The VS_016_DUT_SIPv4 Send - Receive MESSAGE with Registration test scenario is completely configured. Channel#0: MakeRegistration - First Loop Only, Send MESSAGE, Wait 100 Trying or 200 OK for MESSAGE Channel#1: MakeRegistration - First Loop Only, Wait MESSAGE, Send 200 OK for MESSAGE.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The Receive_Call activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page. The SIP server you are running this test against needs specified in the Use external server area. Outbound proxy and registrar functionality on the specified proxy need to be configured.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform any RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** Receive_Call uses the same settings as Make_Call, except for the Dial Plan page, which does not need to specify a destination activity.

 **Note:** The SIP sample templates include another VS_042_DUT_SIPv6 Send - Receive MESSAGE with Registration predefined test identical to this one, except that it uses IPv6 network settings.

VS_017_DUT_SIPv4 IMS MakeCall - ReceiveCall with Registration and RTP

This IMS compliant test runs against a P-CSCF whose IP address needs configured in the **SIP Settings** page. The test comprises two activities, Make_Call and Receive_Call, emulating SIP endpoints that establish an IMS-compliant call.





Make_Call establishes an IMS-compliant call, performs a voice session, and initiates call termination. Receive_Call executes the corresponding functions flow on the receiving side.

The Make_Call settings are described in the following table:

Category	Settings
Scenario	<p>The VS_017_DUT_SIPv4 IMS MakeCall - ReceiveCall with Registration and RTP test scenario comprises two channels:</p> <p>Channel#0: IMS MakeRegistration, IMS Make Call, Voice Session, IMS EndCall Initiate.</p> <p>Channel#1: IMS MakeRegistration, IMS Receive Call, Voice Session, IMS EndCall Terminate.</p>

Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) <p>For each media channel, a unique (IP, port) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • UDP port: Use same value
Dial Plan	Receive_Call activity is configured as call destination.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page. The P-CSCF you are running this test against must be configured in the Use external server area. The authentication settings needs configured in the UAC Authentication area.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test performs media streaming, the Enable media on this activity option is selected.
Other Settings	<p>The IP version preference is set to IPv4 and the following scenario variables need to be initialized:</p> <ul style="list-style-type: none"> • VoIP_Var1: This variable, defined as [310000-], initializes the SIP_Private_Id variable. • VoIP_Var2: This variable, defined as [310000-], initializes the SIP_Private_Id_Pwd variable.

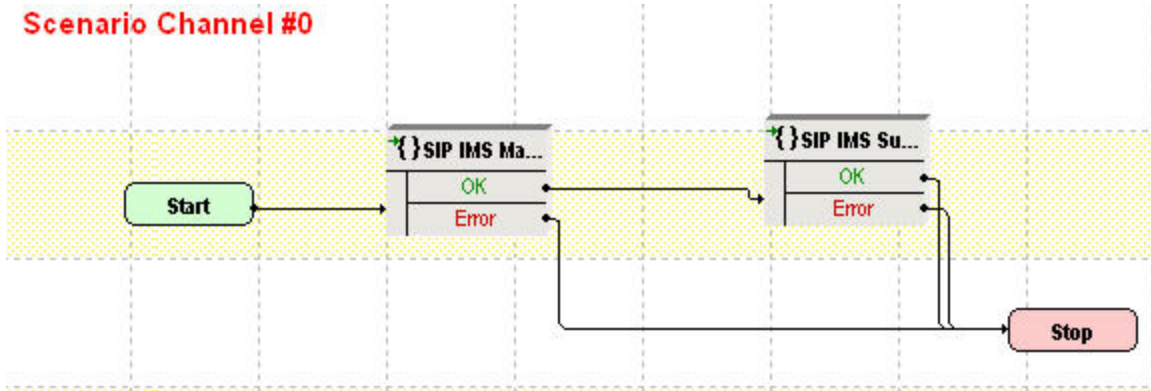
 **Note:** Receive_Call activity is configured similar to Make_Call, except that it does not specify a call destination since it only terminates a call. Similar to the Make_Call-emulated endpoints, the VoIP_Var1 and VoIPVar2 variables of the Other Settings configuration page are also to initialize the SIP_Private_Id and SIP_Private_Id_Pwd variables.

 **Note:** This test also uses scenario variables to assert the private identity of the SIP UAs, as documented in detail in the notes attached to the test scenario.

VS_018_DUT_SIPv4 IMS Registration with Subscription

This IMS-compliant test runs against a P-CSCF server whose IP address needs configured in the **SIP Settings** page.

The test comprises only one VoIPSIPPeer activity, Make_Call, that is linked to a scenario channel executing a registration with a SIP server and a subscription operation, as shown in the following image:



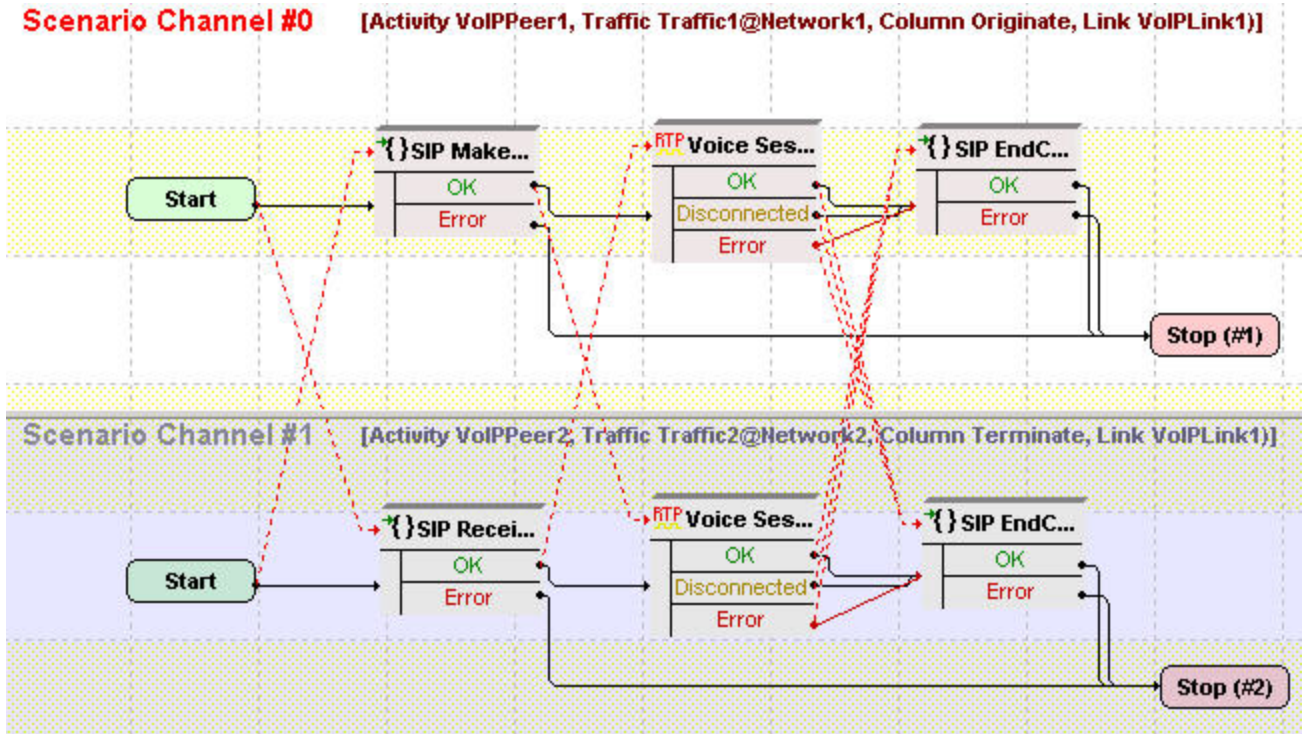
The Make_Call settings are described in the following table:

Category	Settings
Scenario	The VS_018_DUT_SIPv4 IMS Registration with Subscription test scenario comprises 1 channel. Channel#0: IMS MakeRegistration, IMS Subscribe.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	No VoIPSIPPeer activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port = Use same value setting from the Execution Settings page. The P-CSCF you are running this test against must be configured in the Use external server area. The authentication settings needs configured in the UAC Authentication area.
Codec Settings	The codec settings can be left unaltered.
RTP Settings	Because this test does not use RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4 and the following scenario variables need to be initialized: <ul style="list-style-type: none"> • VoIP_Var1: This variable, defined as [310000-], initializes the SIP_Private_Id variable. • VoIP_Var2: This variable, defined as [310000-], initializes the SIP_Private_Id_Pwd variable.

VS_019_DUT_SIPv4 MakeCall - ReceiveCall with RTP - SBC Testing

This test runs against a Session Border Controller (SBC) and comprises two VoIPSIPPeer activities, Make_Call and Receive_Call.

Make_Call is linked to a test scenario channel that registers with a SIP server, establishes a call, performs a voice session, and disconnects, as shown in the following image. Receive_Call executes the test flow on the receiving side.



Note: This test is particular in that it employs a 1-to-N IP mapping, that is, a single IP is used for all signaling and media endpoints emulated by the Make_Call activity, while endpoints emulated by the Receive_Call activity use N IPs.

The Make_Call settings are described in the following table:

Category	Settings
Scenario	<p>The VS_019_DUT_SIPv4 MakeCall - Receive Call with RTP - SBC Testing test scenario comprises two channels.</p> <p>Channel#0: MakeCall - Authentication, Voice Session (33s), EndCall Initiate.</p> <p>Channel#1: ReceiveCall, Voice Session (33s), EndCall Terminate.</p>

Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) <p>For each media channel, a unique (IP, port) tuple is generated using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use same value • UDP port: Use consecutive values (per port)
Dial Plan	The Receive_Call activity is configured as call destination.
SIP Settings	The SIP port has the default 5060 setting. The SBC you are running this test against needs configured in the Use external server area.
Codec Settings	The codec settings can be left unaltered.
RTP Settings	Because this test uses RTP streaming, the Enable media on this activity option is selected. For this activity, whose emulated media channels use the same IP address values, RTP ports need to be configured to the 10000, 10002, 10004, ... sequence.
Other Settings	The IP version preference is set to IPv4 and the following scenario variables need to be initialized.

The Receive_Call settings are described in the following table:

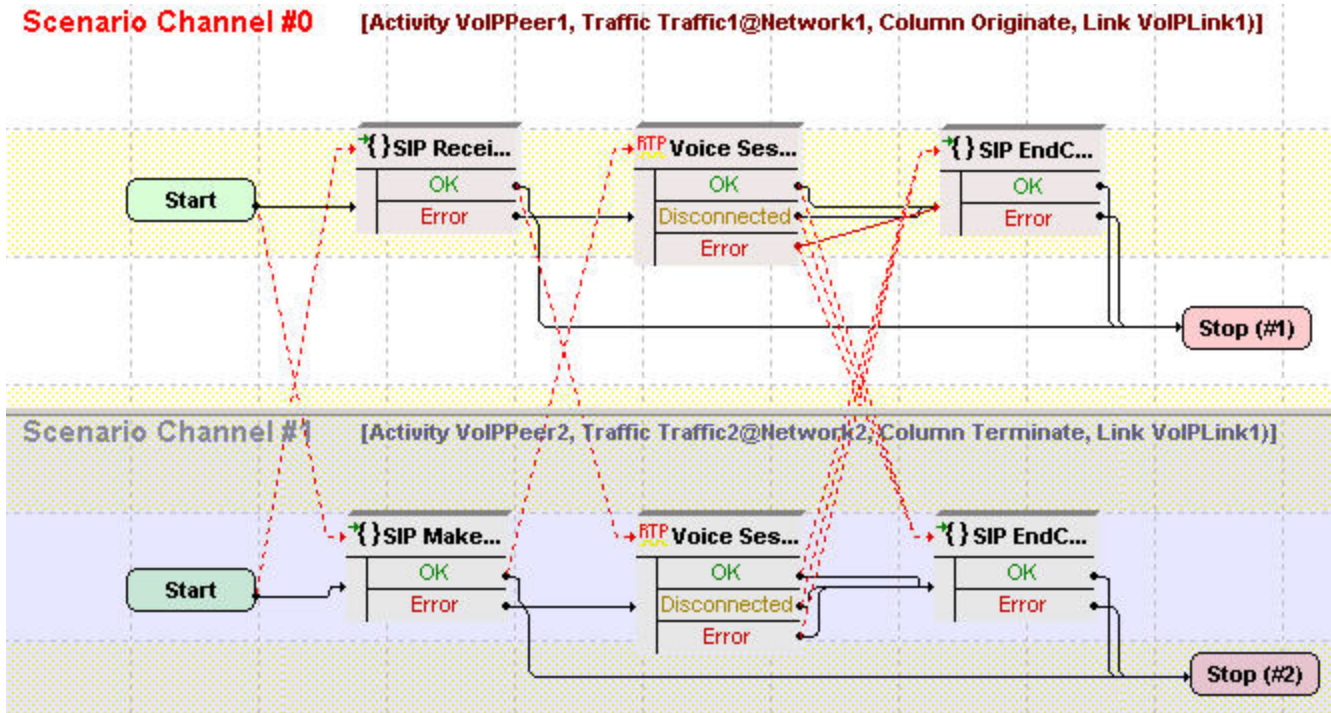
Category	Settings
Scenario	<p>The VS_019_DUT_SIPv4 MakeCall - Receive Call with RTP - SBC Testing test scenario comprises two channels.</p> <p>Channel#0: MakeCall - Authentication, Voice Session (33s), EndCall Initiate.</p> <p>Channel#1: ReceiveCall, Voice Session (33s), EndCall Terminate.</p>
Execution Settings	<p>The channels are configured to use consecutive IPs from those available in the Network settings. For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) <p>For each media channel, a unique (IP, port) tuple is generated using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use same value • UDP port: Use consecutive values (per port)


Dial Plan	No activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port: Use same value setting from the Execution Settings page.
Codec Settings	The codec settings can be left unaltered.
RTP Settings	Because this test uses RTP streaming, the Enable media on this activity option is selected. For this activity, whose emulated channels use multiple, consecutive IP address values, RTP ports can be configured to a single 10000 value.
Other Settings	The IP version preference is set to IPv4 and the following scenario variables need to be initialized.

VS_020_DUT_SIPv4 ReceiveCall - MakeCall with RTP - SBC Testing

This test runs against a Session Border Controller (SBC) and comprises two VoIP SIPPeer activities, Receive_Call and Make_Call.

Receive_Call is linked to a test scenario channel that registers with a SIP server, terminates an incoming call, performs a voice session, and disconnects, as shown in the following image. Make_Call executes the call originating functions flow.



 **Note:** This test uses an N-to-N configuration, that is, Receive_Call and Make_Call both emulate SIP and media endpoints using N distinct IPs.

The Receive_Call settings are given in the following table:

Category	Settings
Scenario Editor	The VS_020_DUT_SIPv4 ReceiveCall - MakeCall with RTP - SBC Testing scenario comprises two channels: Channel#0: ReceiveCall, Voice Session (33s), End Call Initiate Channel#1: MakeCall - Authentication, Voice Session (33s), EndCall Terminate.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) For each media channel, a unique (IP, port) tuple is generated using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use same value • UDP port: Use consecutive values (per port)
Dial Plan	Because this activity terminates a call, no activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port: Use same value setting from the Execution Settings page.
Codec Settings	The codec settings can be left unaltered.
RTP Settings	Because this test uses RTP streaming, the Enable media on this activity option is selected. For this activity, whose emulated channels use multiple, consecutive IP address values, RTP ports can be configured to a single 10000 value.
Other Settings	The IP version preference is set to IPv4 and the following scenario variables need to be initialized.

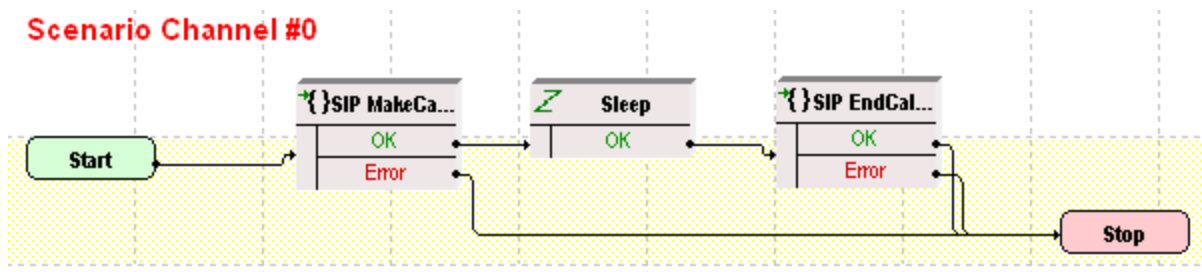
The Make_Call settings are described in the following table:

Category	Settings
----------	----------

Scenario	The VS_020_DUT_SIPv4 ReceiveCall - MakeCall with RTP - SBC Testing scenario comprises two channels: Channel#0: ReceiveCall, Voice Session (33s), End Call Initiate Channel#1: MakeCall - Authentication, Voice Session (33s), EndCall Terminate.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) For each media channel, a unique (IP, port) tuple is generated using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use same value • UDP port: Use consecutive values (per port)
Dial Plan	The Receive_Call activity is configured as call destination.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port: Use same value setting from the Execution Settings page. The SBC you are running this test against is configured in the Use external server area.
Codec Settings	The codec settings can be left unaltered.
RTP Settings	Because this test uses RTP streaming, the Enable media on this activity option is selected. For this activity, whose emulated channels use multiple, consecutive IP address values, RTP ports can be configured to a single 10000 value.
Other Settings	The IP version preference is set to IPv4 and the following scenario variables need to be initialized.

VS_021_DUT_SIPv4 MakeCall - EndCall

This test comprises a single VoIPSIPPeer activity, Make_Call, which is linked to a test scenario channel that establishes a signaling-only call with a user-specified device and then disconnects, as shown in the following image:

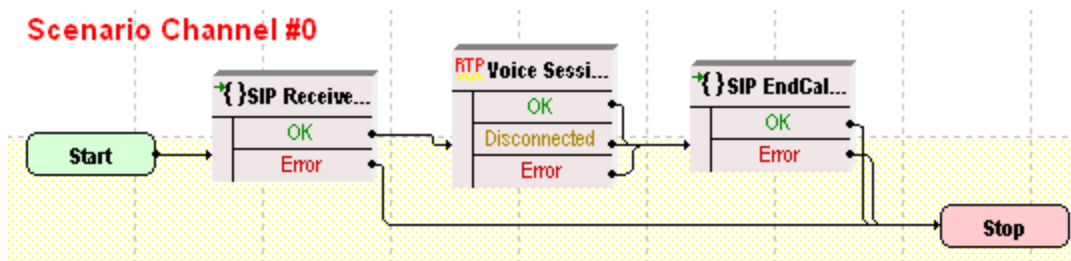


The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario	The VS_021_DUT_SIPv4 MakeCall - EndCall test scenario comprises one channel: Channel#0: MakeCall, Sleep (2s), EndCall
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The IP and port of the call destination device needs to be specified in this page.
SIP Settings	The SIP port has the 5060 default setting.
Codec Settings	The codec settings can be left unaltered.
RTP Settings	Because this test does not perform any RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4 and the following scenario variables need to be initialized.

VS_022_DUT_SIPv4 MakeCall - EndCall with RTP - 33s

This test comprises a single VoIPSIPPeer activity, Make_Call, which is linked to a test scenario channel that establishes a SIP call with media streaming with a user-specified device and then disconnects, as shown in the following image:



The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario	The VS_022_DUT_SIPv4 MakeCall - EndCall with RTP - 33s test scenario comprises one channel: Channel#0: MakeCall, Voice Session, EndCall Initiate.

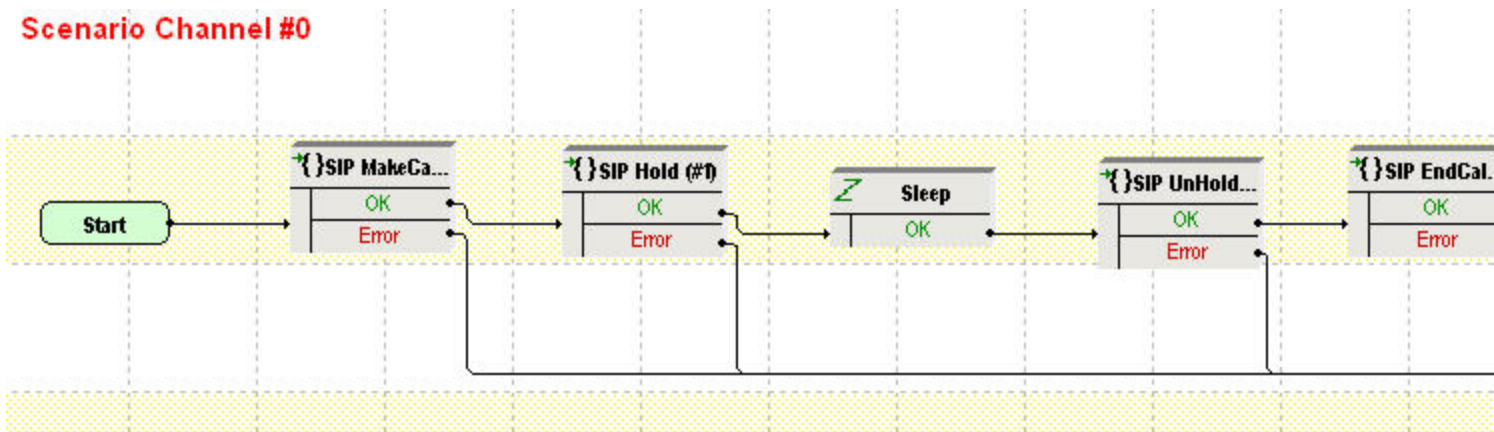
Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) <p>For each media channel, a unique (IP, port) tuple is generated using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • UDP port: Use same value
Dial Plan	The IP and port of the call destination device needs to be specified in this page.
SIP Settings	The SIP port has the 5060 default setting.
Codec Settings	The codec settings can be left unaltered.
RTP Settings	The Enable media on this activity option is selected. For this activity, which uses consecutive values for the media IP addresses, RTP ports for all channels are configured to a single 10000 value.
Other Settings	The IP version preference is set to IPv4 and the following scenario variables need to be initialized.

The SIP sample templates include a VS_042_DUT_SIPv6 MakeCall - EndCall with Voice - 33s predefined test identical to this one, except that it uses IPv6 network settings.

VS_023_DUT_SIPv4 MakeCall - EndCall with Hold Unhold

This test is similar to the previous [VS_021_DUT_SIPv4 MakeCall - EndCall](#) test, except that it executes two additional script functions, SIP Hold and SIP Unhold, before call termination.

Make_Call is linked to the test scenario channel as shown in the following image:



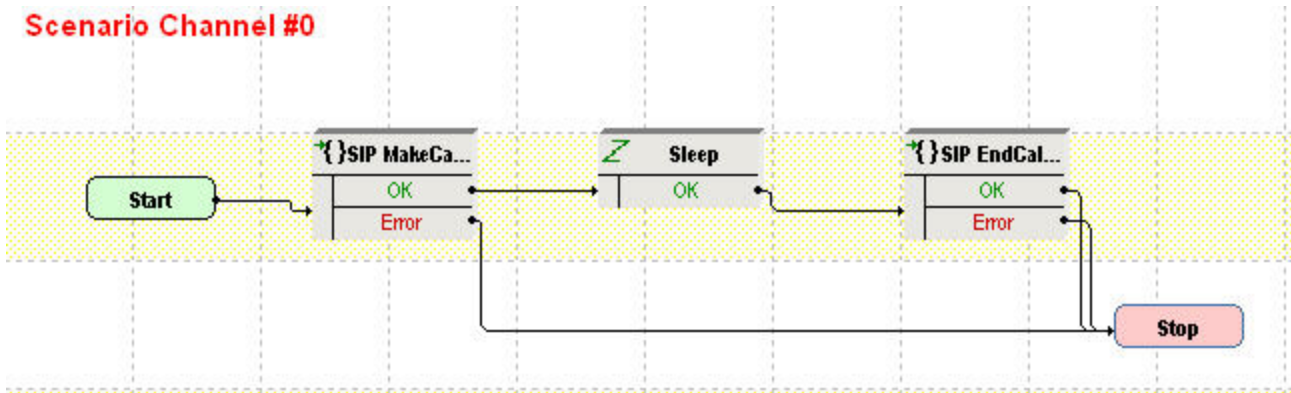
Note: The test configuration is the same as that of the previous [VS_021_DUT_SIPv4 MakeCall - EndCall](#) test.

VS_024_DUT_SIPv4 MakeCall - EndCall with SRTP - 33s

This test is similar to the previous [VS_022_DUT_SIPv4 MakeCall - EndCall with RTP - 33s](#) test, except that the media traffic is encrypted using SRTP.

VS_025_DUT_SIPv4 MakeCall - EndCall through SIP Redirect Server

This test comprising a single VoIPSIPPeer activity illustrates a call establishment procedure through a SIP redirect server. Make_Call is linked to a test scenario channel that establishes a call by using the SIP MakeCall - Redirect Server procedure capable of resolving 3xx messages and by using the first Contact address provided by the redirect server.



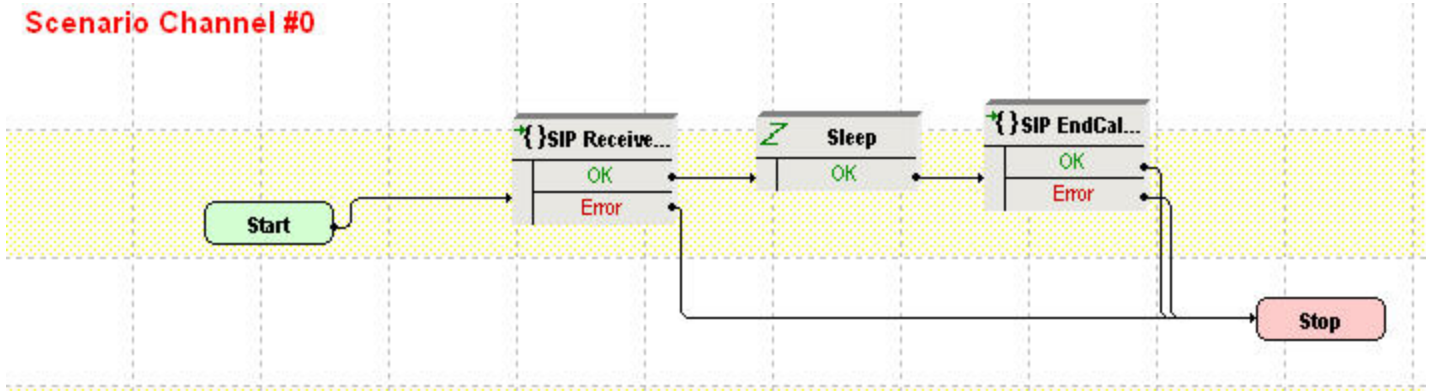
The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario	The VS_025_DUT_SIPv4 MakeCall - EndCall through Redirect Server test scenario comprises one channel: Channel 0: MakeCall - Redirect Server, Sleep, EndCall Initiate.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The IP and port of the call destination device needs to be specified in this page.
SIP Settings	The SIP port has the 5060 default setting for all channels.
Codec Settings	The codec settings can be left unaltered, because no media streaming is performed.

RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4 and the following scenario variables need to be initialized.

VS_026_DUT_SIPv4 ReceiveCall - EndCall

This test comprises a single receiving-side VoIPSIPPeer activity linked to a test scenario that answers an incoming signaling-only call and then disconnects, as shown in the following image:



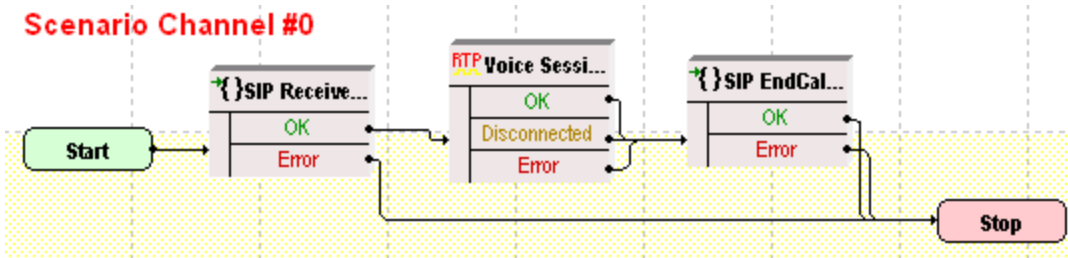
The Receive_Call configuration settings are described in the following table:

Category	Settings
Scenario	The VS_026_DUT_SIPv4 Receive Call - EndCall test scenario comprises one channel: Channel#0: ReceiveCall, Sleep (2s), EndCall Terminate.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	No activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port = Use same value setting from the Execution Settings page. The SIP server you are running this test against needs configured in the Use external server area.
Codec Settings	The codec settings can be left unaltered.

RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4 and the following scenario variables need to be initialized.

VS_027_DUT_SIPv4 ReceiveCall - EndCall with RTP - 33s

This test comprises a single receiving-side VoIPSIPPeer activity linked to a test scenario that answers an incoming call, performs media streaming by using the Voice Session script function and then disconnects, as shown in the following image:



The Receive_Call configuration settings are described in the following table:

Category	Settings
Scenario	The VS_027_DUT_SIPv4 Receive Call - EndCall with RTP - 33s comprises 1 channel: Channel#0: SIP Receive Call, Voice Session, SIP End Call Receive.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) For each media channel, a unique (IP, port) tuple is generated using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • UDP port: Use same value
Dial Plan	No activity is configured as destination activity.
SIP Settings	The SIP port has the 5060 default setting for all channels. The SIP server you are running this test against needs configured in the Use external server area.
Codec Settings	The default codec settings are used.

RTP Settings	The Enable media on this activity option is selected. The RTP port is configured to the same 10000 value for all channels.
Other Settings	The IP version preference is set to IPv4 and the following scenario variables need to be initialized.

Note: The SIP sample templates include another VS_045_DUT_SIPv6 ReceiveCall - EndCall with RTP - 33s predefined test identical to this one, except that it uses IPv6 network settings.

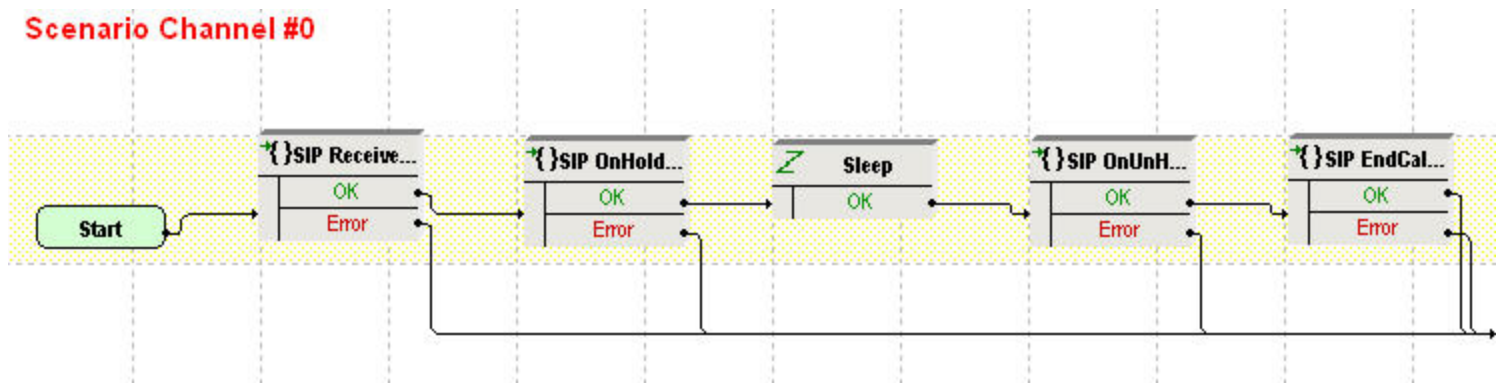
VS_028_DUT_SIPv4 ReceiveCall - EndCall with SRTP - 33s

This test is similar to the previous [VS_027_DUT_SIPv4 ReceiveCall - EndCall with RTP - 33s](#) test, except the media traffic is encrypted by using SRTP.

VS_029_DUT_SIPv4 ReceiveCall - EndCall with Hold Unhold

This test is similar to the previous [VS_026_DUT_SIPv4 ReceiveCall - EndCall](#) test, except that it uses another two script functions, SIP Hold and SIP Unhold, before call termination.

Receive_Call is linked to the test scenario channel as shown in the following image:



The test configuration is the same as that of the previous [VS_026_DUT_SIPv4 ReceiveCall - EndCall](#) test.

VS_030_B2B_SIPv4_TLS_MakeCall - ReceiveCall - EndCall

This test is similar to the previous [VS_001_B2B_SIPv4 MakeCall - ReceiveCall - EndCall](#) test, except that the signaling traffic is encrypted by using TLS. SIP endpoints emulated by the VoIPSIPPeer activities authenticate each other with certificates (.pem format) included with the IxLoad application.

VS_031_B2B_SIPv4_TLS_MakeCall - ReceiveCall - EndCall with RTP - 33s


This test is similar to the previous [VS_002_B2B_SIPv4 MakeCall - ReceiveCall - EndCall with RTP - 33s](#) test, except that the signaling traffic is encrypted by using TLS. SIP endpoints emulated by the VoIPSIPPeer activities authenticate each other with certificates (.pem format) included with the IxLoad application.


VS_032_DUT_SIPv4_TLS_MakeCall - ReceiveCall with Registration

This test is similar to the previous [VS_006_DUT_SIPV4_MakeCall - ReceiveCall with Registration](#) test, except that the signaling traffic is encrypted by using TLS. For SIP endpoints authentication, in the **TLS Settings** tab, you need to specify a proper certificate.

VS_033_B2B_SIPV4_TLS_MakeCall - ReceiveCall - EndCall with SRTP - 33s

This test is similar to the previous [VS_003_B2B_SIPV4_MakeCall - ReceiveCall - EndCall with SRTP - 33s](#) test, except that the signaling traffic is encrypted by using TLS. SIP endpoints emulated by the VoIPSIPPeer activities authenticate each other with certificates (.pem format) included with the IxLoad application.

 **Note:** Further sample test configurations with indexes up to VS_050 that are provided in the IxLoad installation kit represent configurations similar to those listed above, except that they are configured using IPv6 instead of IPv4 settings.

 **Note:** The following tests specially created for use with Acceleron load module boards are stored both in the VoIPSIP\Acceleron_NA and the VoIPSIP\Acceleron_10G folders of the IxLoad **Getting Started** window.

SIPV4_UDP_Basic_Call_without_RTP_Max_CPS

This is a performance test that runs in Back-to-Back mode and comprises two VoIPSIPPeer activities, Make_Call and Receive_Call. Make_Call is linked to a test scenario channel that establishes a basic SIP call without media streaming and then disconnects. Receive_Call executes the test flow on the receiving side. The configured test objective is 700 CPS.

SIPV4_TCP_Basic_Call_without_RTP_Max_CPS

This test is similar to the previous [SIPV4_UDP_Basic_Call_without_RTP_Max_CPS](#) test, except that the signaling traffic is sent over TCP. The configured test objective is 600 CPS.

SIPV4_TLS_Basic_Call_without_RTP_Max_CPS

This test is similar to the previous [SIPV4_UDP_Basic_Call_without_RTP_Max_CPS](#) test, except that the signaling traffic is sent by using TLS. The configured test objective is 600 CPS.

SIPV4_UDP_Basic_Call_with_RTP_1sec_Max_CPS

This is a performance test that runs in Back-to-Back mode and comprises two VoIPSIPPeer activities, Make_Call and Receive_Call. Make_Call is linked to a test scenario channel that establishes a basic SIP call with media streaming (by using the Voice Session script function), and then disconnects. Receive_Call executes the test flow on the receiving side. The configured test objective is 500 CPS.

SIPV4_TCP_Basic_Call_with_RTP_1sec_Max_CPS

This test is similar to the previous [SIPV4_UDP_Basic_Call_with_RTP_1sec_Max_CPS](#) test, except that the signaling traffic is sent over TCP. The configured test objective is 500 CPS.

SIPV4_TLS_Basic_Call_with_RTP_1sec_Max_CPS

This test is similar to the previous [SIPV4_UDP_Basic_Call_with_RTP_1sec_Max_CPS](#) test, except that the signaling traffic is sent by using TLS. The configured test objective is 250 CPS.

SIPv4_UDP_Basic_Call_with_RTP_30sec_Max_CPS

This test is similar to the previous [SIPv4_UDP_Basic_Call_with_RTP_1sec_Max_CPS](#) test, except that the media is played for a duration of 30 seconds. The configured test objective is 200 CPS.

SIPv4_TCP_Basic_Call_with_RTP_30sec_Max_CPS

This test is similar to the previous [SIPv4_UDP_Basic_Call_with_RTP_30sec_Max_CPS](#) test, except that the signaling traffic is sent over TCP. The configured test objective is 200 CPS.

SIPv4_TLS_Basic_Call_with_RTP_30sec_Max_CPS

This test is similar to the previous [SIPv4_UDP_Basic_Call_with_RTP_30sec_Max_CPS](#) test, except that the signaling traffic is sent by using TLS. The configured test objective is 200 CPS.

SIPv4_UDP_Basic_Call_with_RTP_3min_Max_CPS

This test is similar to the previous [SIPv4_UDP_Basic_Call_with_RTP_1sec_Max_CPS](#) test, except that the media is played for a duration of three minutes. The configured test objective is 40 CPS.

SIPv4_TCP_Basic_Call_with_RTP_3min_Max_CPS

This test is similar to the previous [SIPv4_UDP_Basic_Call_with_RTP_3min_Max_CPS](#) test, except that the signaling traffic is sent by using TLS. The configured test objective is 40 CPS.

SIPv4_TLS_Basic_Call_with_RTP_3min_Max_CPS

This test is similar to the previous [SIPv4_UDP_Basic_Call_with_RTP_3min_Max_CPS](#) test, except that the signaling traffic is sent over TCP. The configured test objective is 40 CPS.

SIPv4_UDP_Basic_Call_with_RTP_Max_Sessions

This is a performance test that runs in Back-to-Back mode and comprises two VoIPSIPPeer activities, Make_Call and Receive_Call. Make_Call is linked to a test scenario channel that establishes a basic SIP call with media streaming (by using the Voice Session script function), and then disconnects. Receive_Call executes the test flow on the receiving side. The configured test objective is 8000 Channels.

SIPv4_TCP_Basic_Call_with_RTP_Max_Sessions

This test is similar to the previous [SIPv4_UDP_Basic_Call_with_RTP_Max_Sessions](#) test, except that the media is sent by using SRTP. The configured test objective is 8000 Channels.

SIPv4_TLS_Basic_Call_with_RTP_Max_Sessions

This test is similar to the previous [SIPv4_UDP_Basic_Call_with_RTP_Max_Sessions](#) test, except that the media is sent by using SRTP. The configured test objective is 8000 Channels.

SIPv4_UDP_Basic_Call_with_SRTP_Max_Sessions

This test is similar to the previous [SIPv4_UDP_Basic_Call_with_RTP_Max_Sessions](#) test, except that the media is sent by using SRTP. The configured test objective is 300 Channels.

SIPv4_TCP_Basic_Call_with_SRTP_Max_Sessions

This test is similar to the previous [SIPv4_TCP_Basic_Call_with_RTP_Max_Sessions](#) test, except that the media is sent by using SRTP. The configured test objective is 300 Channels.

SIPv4_TLS_Basic_Call_with_SRTP_Max_Sessions

This test is similar to the previous [SIPv4_TLS_Basic_Call_with_RTP_Max_Sessions](#) test, except that the media is sent by using SRTP. The configured test objective is 200 channels.

Note: The following two tests are stored in the VoIPSIP\Proxy_Scenarios folder of the IxLoad **Getting Started** window.

SIPv4_Proxy

This test illustrates the case of SIP UAs that register with a Registrar, and then establishes calls through an SIP Proxy server, whereby all test entities – SIP UAs, Registrar and Proxy server – are emulated by IxLoad activities. During the call establishment phase, the Proxy that stays is the message path, while after the call is established, media is exchanged directly between SIP endpoints.

Caller and Callee endpoints are emulated by the Make_Call and Receive_Call VoIPSIPPeer activities; the Proxy/Registrar is simulated by four VoIPSIPPeer activities having an associated VoIPSIPCloud activity:



The executed protocol flow is the following:

- Caller executes the SIP protocol flow of the calling endpoints, including the endpoint registration/de-registration operations at the start and the end of the test flow. On the established call, a Talk script function plays media directly between Caller and Callee. Finally Caller terminates the call using an End Call Initiate procedure.
- Callee executes the SIP protocol flow of the calling endpoints, including the endpoint registration/de-registration operations at the start and the end of the test flow. On the established call, a Talk script function plays media directly between Callee and Caller.
- R_Caller, R_Callee handle the registration/de-registration for the Caller and Callee endpoints respectively.
- Wait_Call handles the incoming call from the Proxy server perspective by communicating with the Caller: an initial INVITE Processing procedure waits for an incoming INVITE message, then sends the response by using a Send 180 and 200 Ok procedure.

Note that inside the INVITE Processing procedure, the Wait INVITE procedure extracts the value of some message headers, which are then used by the subsequent Variable Set function to

initialize a number of scenario variables. These variables are eventually used by the Send INVITE procedure of the Make_Call activity for initializing the values of INVITE message headers.

Using a Wait procedure, the scenario execution is then paused for a duration of 1 second, providing a time window for the SIP UAs to exchange media directly, without the Proxy staying in the media path.

Finally, a SIP EndCall - Receiving Channel procedure waits for an incoming BYE message from Caller, sets the E(nd)C(all)R(eceived) scenario variable to '1' ('true'), and responds with a 200 Ok message. In case no BYE message was received, the SIP EndCall - Receiving Channel procedure exits on the Timeout output and checks whether a BYE message was received by the Make_Call activity (test scenario channel 2) by testing the ECR value for that channel.

- Make_Call handles the outgoing call from the Proxy server perspective by communicating with the Callee: When an INVITE message is received by Wait_Call, this is retransmitted by using a Send INVITE procedure, while subsequent procedures wait for responses to the INVITE and send an ACK.

Note that the Send INVITE procedure uses a number of scenario variables initialized within the INVITE Processing procedure for passing appropriate values to the message headers of the sent INVITE message.

Using a Wait procedure, the scenario execution is then paused for a duration of 1 second, providing a time window for the SIP UAs to exchange media directly, without the Proxy staying in the media path.

Finally, a SIP EndCall - Calling Channel procedure waits for a BYE message, sets the E(nd)C(all)R(eceived) scenario variable to '1' ('true'), and completes the call. In case no BYE message was received, the SIP EndCall - Calling Channel procedure exits on the Timeout output and checks whether a BYE message was received by the Wait_Call activity (test scenario channel 1) by testing the ECR value for that channel.

SIPv4_B2BUA

This test is similar to the previous [SIPv4_Proxy](#) test, except that it emulates a B2B User Agent (UA) instead of a Proxy server.

VoIPSIP with MSRP

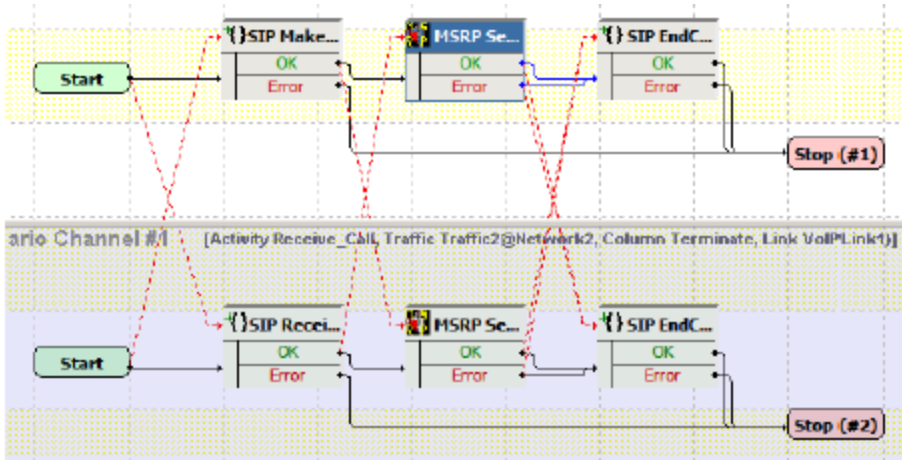
The tests from this category emulate SIP UAs that establish SIP sessions, and then exchange text or transfer files by using the MSRP protocol. Some tests are also provided that transmit voice over SIP calls and simultaneously negotiate other SIP sessions for the sending of text or files by using MSRP.

All tests from this category run in Back-to-Back mode and contain two VoIPSIPPeer activities, Make_Call and Receive_Call.

MSRP_01_SIPv4_UDP_Text_Bidirectional

Make_Call is linked to a test scenario channel that establishes a SIP call, sends a text message by using the MSRP Session script function, and then disconnects as shown in the following image. Receive_Call executes the test flow on the receiving side, whereby it uses a corresponding MSRP

Session script function to receive and also send a text message over the established session (bidirectional text transfer).



The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario	The test scenario comprises two channels executing a basic call procedure with media exchange. Channel#0: MakeCall procedure, MSRP Session function, End Call Initiate procedure. Channel#1: ReceiveCall procedure, MSRP Session function, End Call Receive procedure.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The source phone numbers are specified by using the 160[00000000-] regular expression. The Receive_Call activity and port 5060 are configured as call destination.
SIP Settings	The 5060 default SIP listening port setting is configured for all channels.
Codec Settings	The default codec settings are used.
RTP Settings	Because no RTP media is streamed by the emulated SIP endpoints, the Enable media on this activity option is not selected.

MSRP settings	The MSRP endpoints are specified by using the 160[00000000-].example.com sequence generator expression. The text message to be transmitted is configured in the Content page of the MSRP Session function.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity, because it only terminates a call. The MSRP Session function that is executed by the Receive- Call activity also specifies a text message to be transmitted.

MSRP_02_SIPv4_UDP_File_Transfer_Uni_aSDP


Make_Call is linked to a test scenario channel that establishes a SIP call and negotiates file transfer parameters using a SIP Make Call procedure, sends a synthetic file by using the MSRP Session script function, and then disconnects by using a SIP End Call Initiate procedure, as shown in the preceding image. The INVITE function contained in the SIP Make Call procedure uses an automatic SDP (aSDP) offer.

Receive_Call executes a similar test flow on the receiving side, whereby its MSRP Session script function only receives the incoming data (unidirectional file transfer).

The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario	The test scenario comprises two channels executing a basic call procedure with media exchange. Channel#0: MakeCall procedure, MSRP Session function, End Call Initiate procedure. Channel#1: ReceiveCall procedure, MSRP Session function, End Call Receive procedure.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The source phone numbers are specified by using the 160[00000000-] regular expression. The Receive_Call activity and port 5060 are configured as call destination.
SIP Settings	The 5060 default SIP listening port setting is configured for all channels.
Codec Settings	The default codec settings are used.

RTP Settings	Because no RTP media is streamed by the emulated SIP endpoints, the Enable media on this activity option is not selected.
MSRP settings	The MSRP endpoints are specified by using the 160[00000000-].example.com sequence generator expression. The transmitted synthetic file is configured in the Content page of the MSRP Session function.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity, because it only terminates a call.

MSRP_03_SIPv4_UDP_File_Transfer_Uni_customSDP

Make_Call is linked to a test scenario channel that establishes a call using a SIP Make Call procedure, sends a synthetic file using the MSRP Session script function, and then disconnects by using a SIP End Call Initiate procedure, as shown in the preceding image. The INVITE function contained in the SIP Make Call procedure uses a custom SDP (cSDP) offer, whereby the SDP file-selector attribute is defined by using the predefined VoIP_MSRRPFile0, VoIP_MSRRPFileType0, VoIP_MSRRPFileSize0, and VOIP_MSRRPFileHash0 variables, which specify the transmitted file configured on the MSRP Session function. Another SDP attribute, file-transfer-id, is initialized by using the Ixload predefined generateguid function.

Receive_Call executes a similar test flow on the receiving side, whereby its MSRP Session script function only receives the incoming file (unidirectional transfer).

The activity-level settings for this test are the same as those for the previous [MSRP_02_SIPv4_UDP_File_Transfer_Uni_aSDP](#) test.

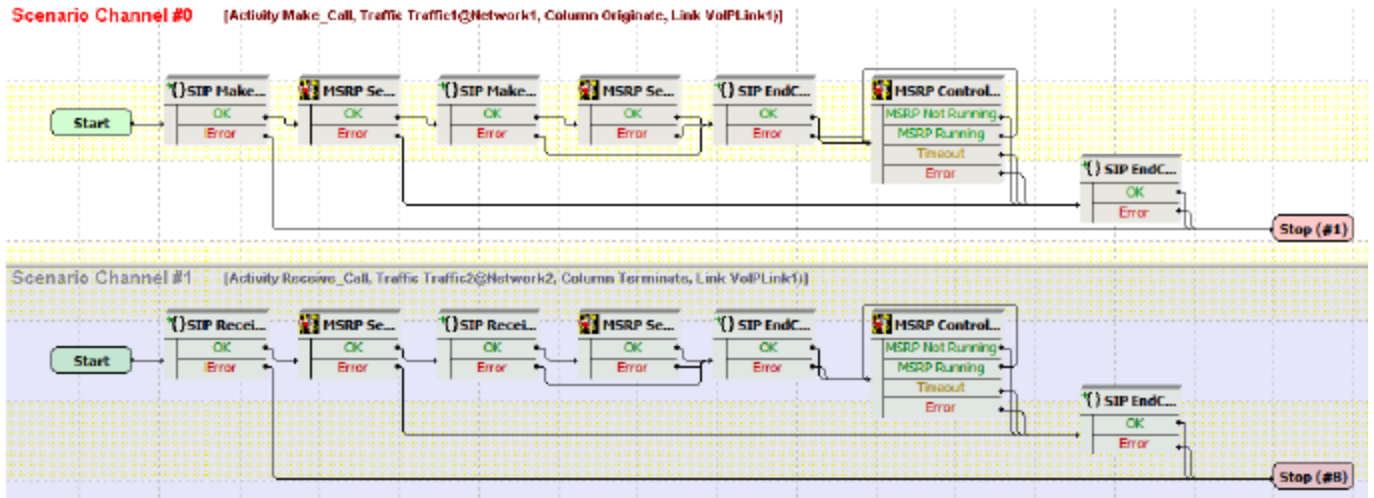
MSRP_04_SIPv4_UDP_Simultaneous_File_Text_Transfer

This test performs the simultaneous sending of a text message and of a synthetic file from the Make_Call activity to the Receive_Call activity.

For each media to be transmitted, text and file, the Make_Call activity executes SIP Make Call procedures that establish the call and negotiate media parameters, and then different MSRP Session functions are called for transmitting the file and the text.

Following the first SIP Make Call which initiates the session for file transfer, a MSRP Session function is executed that sends a synthetic file. This MSRP Session function is configured in a non-blocking mode (background execution mode), so execution advances to a second SIP Make Call procedure, which initiates the session for the sending of text using a second MSRP Session function. This results in the two MSRP Session functions transmitting simultaneously. A first SIP End Call procedure eventually terminates the session for the sending of text.

Because the first MSRP Session function is configured in non-blocking mode, a MSRP Control script function is used to probe if the file transfer is completed or not. Eventually, when the MSRP file transmission is found to have completed, a second SIP End Call is called for terminating the file transfer session. This function's flow is illustrated in the following image:



Receive_Call executes a similar test flow on the receiving side, whereby its MSRP Session script functions receives the incoming text and file data.

MSRP_05_SIPv4_UDP_Simultaneous_Voice_Text_Transfer

This test performs the simultaneous sending of audio (voice) and of a text message from the Make_Call activity to the Receive_Call activity.

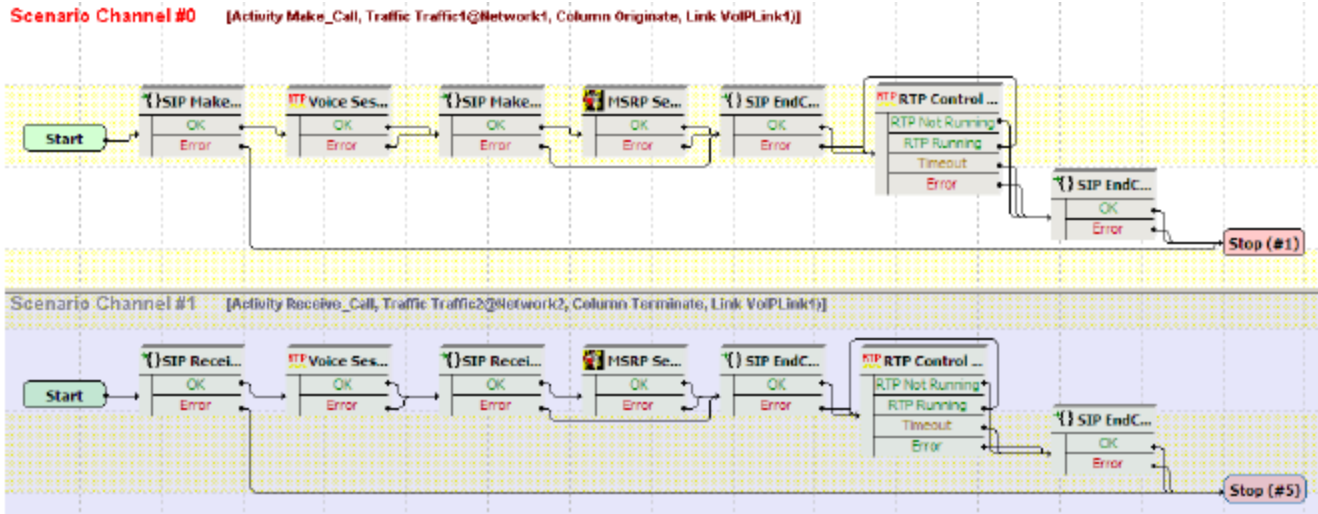
For each media to be transmitted, voice and text, the Make_Call activity executes two SIP Make Call procedures, whereby each contained INVITE function sends a media-specific custom SDP offer using VoIP predefined variables. After the SIP calls have been established, Voice Session and MSRP Session functions are called for transmitting the voice and the text simultaneously.

Following the first SIP Make Call that initiates the SIP session for speech, a Voice Session function is executed that plays an audio file. This Voice Session function is configured in a non-blocking mode (background execution mode) at activity level, so execution advances to a second SIP Make Call procedure, which initiates the session for the sending of text using a MSRP Session function. This results in the Voice Session and the MSRP Session functions transmitting simultaneously. A first SIP End Call procedure eventually terminates the session for the sending of text by using MSRP.

Because the Voice Session function is configured in non-blocking mode at activity level, a RTP Control script function is used to probe if the transmission of speech completed or not; eventually, when the sending of speech is found to have completed, a SIP End Call Initiate is executed for terminating the voice session.

The Receive_Call activity executes a similar test flow on the receiving side, whereby its MSRP Session script function both receives and sends a text message over the same established session.

This functions flow is illustrated in the following image:




Receive_Call executes a similar test flow on the receiving side, whereby its Voice Session function receives the voice data sent by Make_Call, while the MSRP Session function receives the incoming text data.

The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario	<p>The test scenario comprises two channels executing the aforementioned functions flow:</p> <p>Channel#0: SIP Make Call procedure, Voice Session function, SIP Make Call procedure, MSRP Session function, EndCall Initiate procedure, RTP Control function, EndCall Initiate function.</p> <p>Channel#1: SIP Receive Call procedure, Voice Session function, SIP Receive Call procedure, MSRP Session function, EndCall Receive procedure, RTP Control function, EndCall Receive function.</p>
Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	<p>The source phone numbers are specified by using the 160[00000000-] regular expression. The Receive_Call activity and port 5060 are configured as call destination.</p>
SIP Settings	<p>The 5060 default SIP listening port setting is configured for all channels.</p>

Codec Settings	The default G711 aLaw and uLaw codecs are used.
RTP Settings	The Enable media on this activity option is selected for media streaming to be performed. The non-blocking behavior is selected for RTP Session script function.
MSRP settings	The MSRP endpoints are specified by using the 160[00000000-].example.com sequence generator expression.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.


 **Note:** Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity, because it only terminates a call.

MSRP_06_SIPv4_UDP_Simultaneous_Voice_File_Transfer

This test is similar to the [MSRP_05_SIPv4_UDP_Simultaneous_Voice_Text_Transfer](#) one, with the difference that the MSRP Session function from channel#0 transmits a synthetic file instead of a text message. The corresponding MSRP Session function from channel#1 receives the file and does not transmit anything.

VoIPSIP Cloud Test Configurations

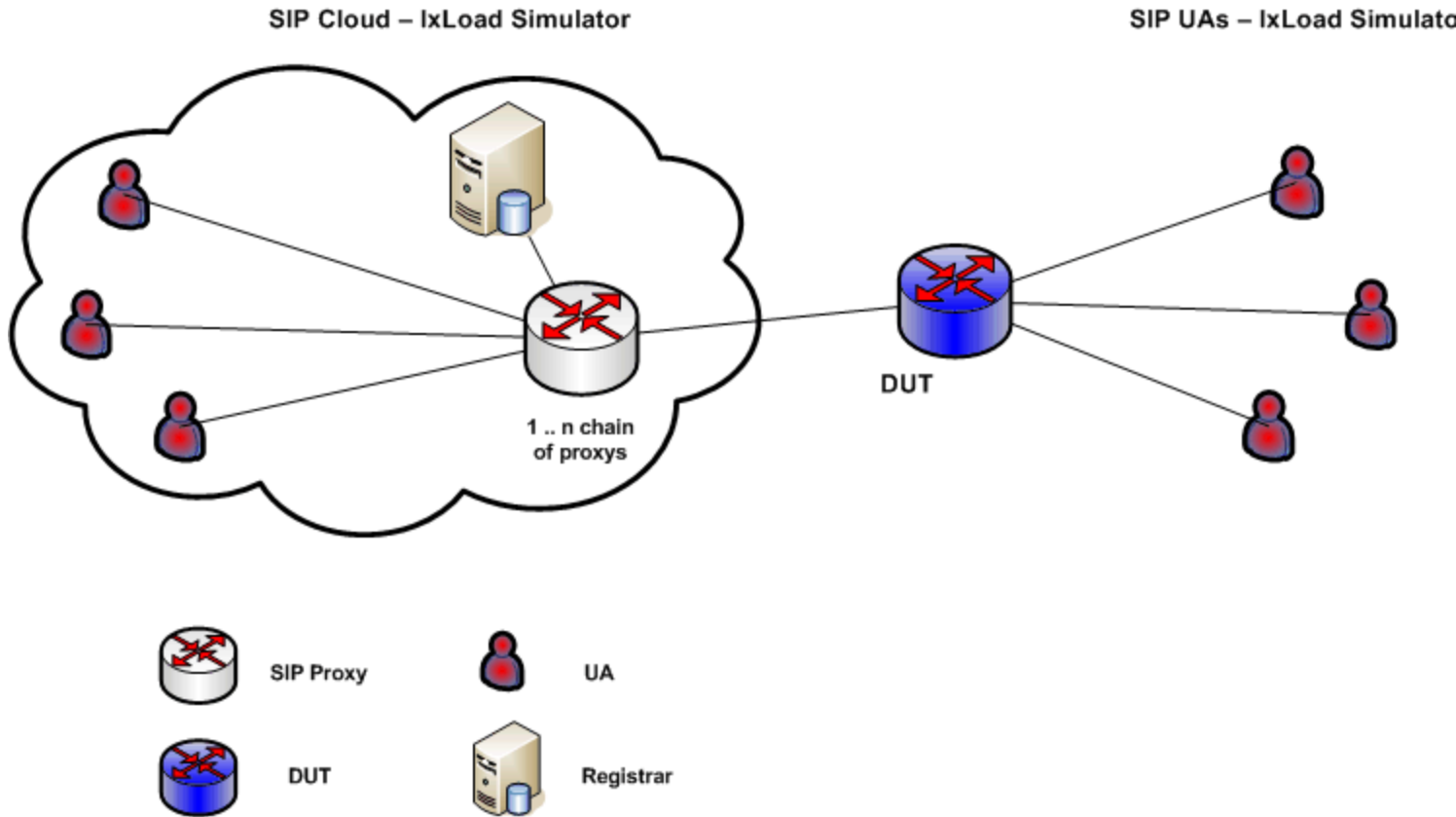
This section provides a brief description of the sample VoIPSIP Cloud test configuration files contained in the IxLoad installation kit.

 **Note:** Sample SIP test configurations do not have retransmissions enabled at activity level. For complex tests that require retransmissions you can enable retransmissions at activity level.

For these tests, VoIPSIP Peer activities are assigned to VoIPSIP Clouds that emulate a number of SIP Proxy servers.

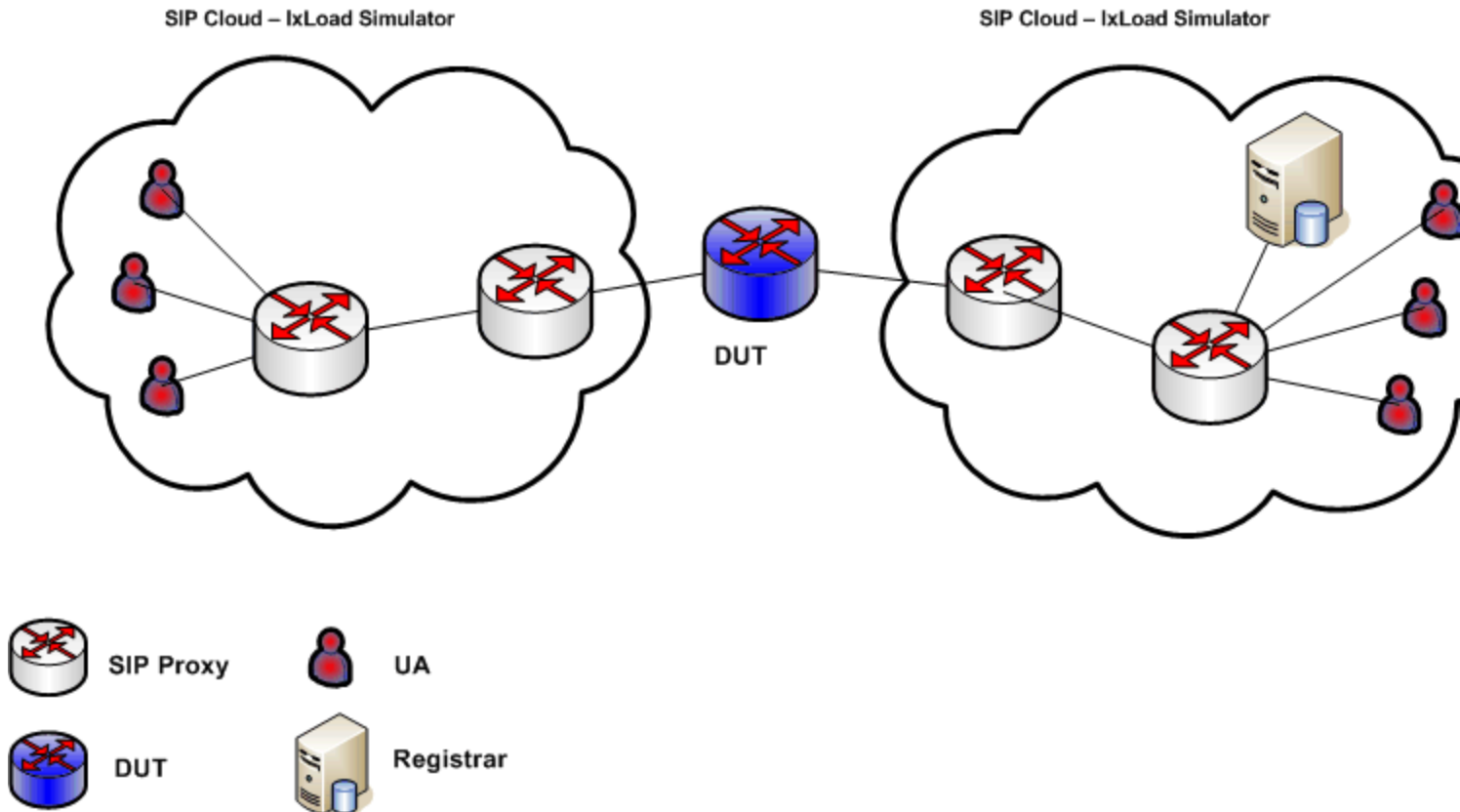
VoIPSIP Cloud test configurations are based on the topologies shown in the following two images:

Topology 1



In this topology, SIP UAs on one side, emulated by a VoIPSIP Peer activity, are configured as part of a SIP cloud, whereas remote UAs are not part of a cloud.

Topology 2

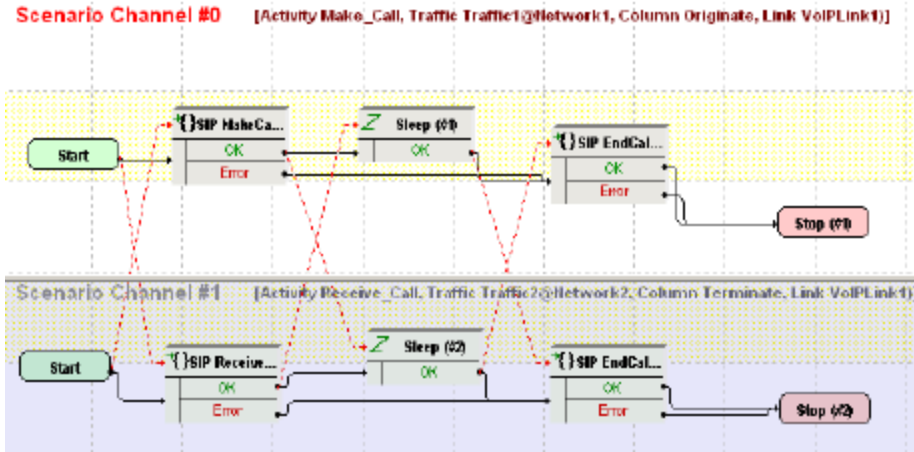


In this topology, both the SIP UAs on the call originating side and those on the call terminating side, emulated by VoIPSIP Peer activities, are configured into a SIP cloud.

SC_001_B2B_SIPv4_T1_MakeCall_from_Cloud

This test, based on the [preceding first topology](#), comprises a VoIPSIP Peer and a VoIPSIPCloud activity on the call originating side and a VoIPSIP Peer activity on the terminating side. The DUT shown in the [preceding first topology](#) is actually not present in the test configuration, the test being run in Back-to-Back mode.

The SIP endpoints emulated by Make_Call and configured into the SIP cloud establish signaling-only calls with the endpoints emulated by Receive_Call.



The Make_Call configuration settings are described in the following table:

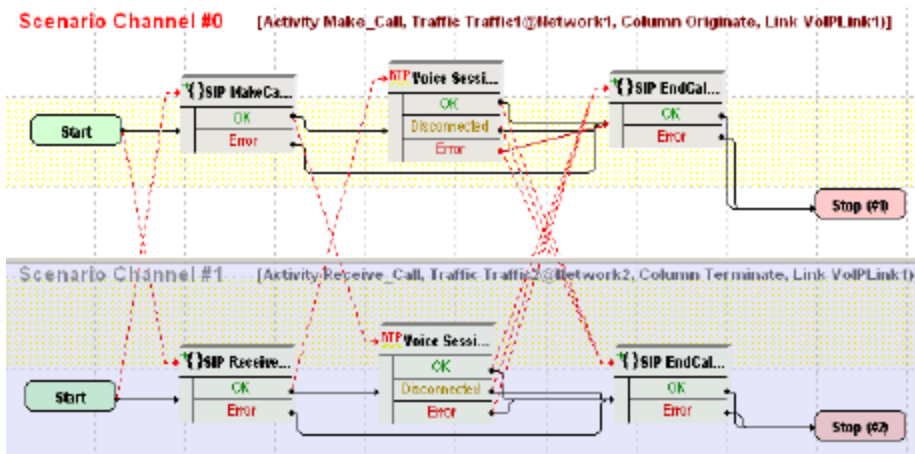
Category	Settings
Scenario Editor	The SC_001_B2B_SIPv4_T1_BasicCall test scenario comprises a basic call originating sequence on the first channel, and a call terminating sequence on the second channel.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The Receive_Call activity is configured as call destination.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page.
Cloud SIP Settings	Make_Call is assigned to VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with Use Server (servers are included in the initial messages path) and Keep in Route (servers remain in the subsequent messages path) options selected. No overriding dispatching rules are defined.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform any RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity. In addition, Receive_Call does not use SIP cloud emulation.

SC_002_B2B_SIPv4_T1_MakeCall_from_Cloud_RTP_n_to_n

This test replicates the [preceding first topology](#) and comprises a VoIPSIP Peer and a VoIPSIPCloud activity on the call originating side, and a VoIPSIPPeer activity on the terminating side. The DUT shown in the [preceding first topology](#) is actually not present in the test configuration, the test being run in Back-to-Back mode.


The SIP UAs emulated by Make_Call and configured in the SIP cloud establish signaling and media calls with the endpoints emulated by Receive_Call.



The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_002_B2B_SIPv4_T1_MakeCall_from_Cloud_RTP_n_to_n test scenario comprises a basic call originating functions sequence on the first channel, and a call terminating sequence on the second channel. After call establishment, media is exchanged directly between the endpoints by using the VoiceSession function.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) For each media channel, a unique (IP, port) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • UDP port: Use same value

Dial Plan	The Receive_Call activity is configured as call destination.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page.
Cloud SIP Settings	Make_Call is assigned to VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with Use Server (servers are included in the initial messages path) and Keep in Route (servers remain in the subsequent messages path) options selected. No overriding dispatching rules are defined.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is selected. The RTP port uses the same 10000 value for all channels.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity. In addition, Receive_Call does not use SIP cloud emulation.

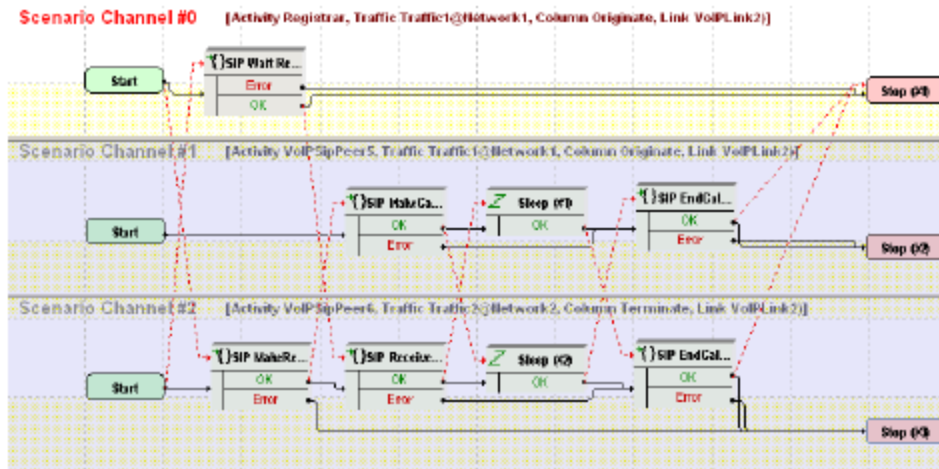
SC_003_B2B_SIPv4_T1_MakeCall_from_Cloud_RTP_1_to_n

This test is similar to the previous SC_002_B2B_SIPv4_T1_MakeCall_from_Cloud_RTP_n_to_n test, except that media endpoints on the originating activity are configured by using a single IP address (which differs from the signaling IP addresses) and consecutive UDP ports.

SC_004_B2B_SIPv4_T1_MakeCall_from_Cloud_w_Registration

This test replicating the [preceding first topology](#) comprises two VoIPSIP Peer (Make_Call and Registrar) and a VoIPSIP Cloud activity on the call originating side, and a VoIPSIPPeer activity (Receive_Call) on the terminating side. The DUT shown in the [preceding first topology](#) is actually not present in the test configuration, the test being run in Back-to-Back mode.

After registering with the Registrar server emulated by the Registrar activity, the SIP UAs emulated by the Receive_Call activity terminate incoming signaling only calls originated by the endpoints emulated by the Make_Call activity.




Both the Make_Call and the Registrar VoIPSIP Peer activities are assigned to VoIPSIPCloud1.

The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_004_B2B_SIPv4_T1_MakeCall_from_Cloud_w_Registration test scenario comprises a basic call originating sequence on the second channel, and a call terminating sequence on the third channel. Before receiving the call, terminating endpoints first execute a registration with the Registrar-emulated server. The first channel function flow, linked to the Registrar activity, responds to incoming registration requests sent by endpoints emulated by the Receive_Call activity.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The Receive_Call activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port = Use same value setting from the Execution Settings page.
Cloud SIP Settings	Make_Call is assigned to VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with Use Server (servers are included in the initial messages path) and Keep in Route (servers remain in the subsequent messages path) options selected.
Codec Settings	The default codec settings are used.

RTP Settings	The Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and the VoIP_IPAddress0 variable is initialized to a range of IP addresses that are assigned to endpoints emulated by Receive_Call activity. The variable is used on the Send INVITE script function to specify the call destination.

 **Note:** Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity. In addition, Receive_Call does not use SIP cloud emulation.

The Receive_Call configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_004_B2B_SIPv4_T1_MakeCall_from_Cloud_w_Registration test scenario comprises a basic call originating sequence on the second channel, and a call terminating sequence on the third channel. Before receiving the call, terminating endpoints first execute a registration with the Registrar-emulated server. The first channel function flow, linked to the Registrar activity, responds to incoming registration requests sent by endpoints emulated by the Receive_Call activity.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	No activity is configured as call destination.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page. The registrar server is specified in the Use external server area.
Cloud SIP Settings	This activity is not assigned to any SIP cloud.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

The Registrar configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_004_B2B_SIPv4_T1_MakeCall_from_Cloud_w_Registration test scenario comprises a basic call originating sequence on the second channel, and a call terminating sequence on the third channel. Before receiving the call, terminating endpoints first execute a registration with the Registrar-emulated server. The first channel function flow, linked to the Registrar activity, responds to incoming registration requests sent by endpoints emulated by the Receive_Call activity.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	Phone numbers are specified by using the 170[00000000-] formula. Because this activity only receives incoming registration requests, no call destination needs to be configured.
SIP Settings	The SIP port has the default setting [5070-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page.
Cloud SIP Settings	This activity is assigned to VoIPSIPCloud1. The activity has an overriding dispatching rule defined, specifying that incoming SIP Register request messages be dispatched based on the 170[00000000-] formula. The Use server option is enabled for the sip_server#3 associated with the emulated Registrar server.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

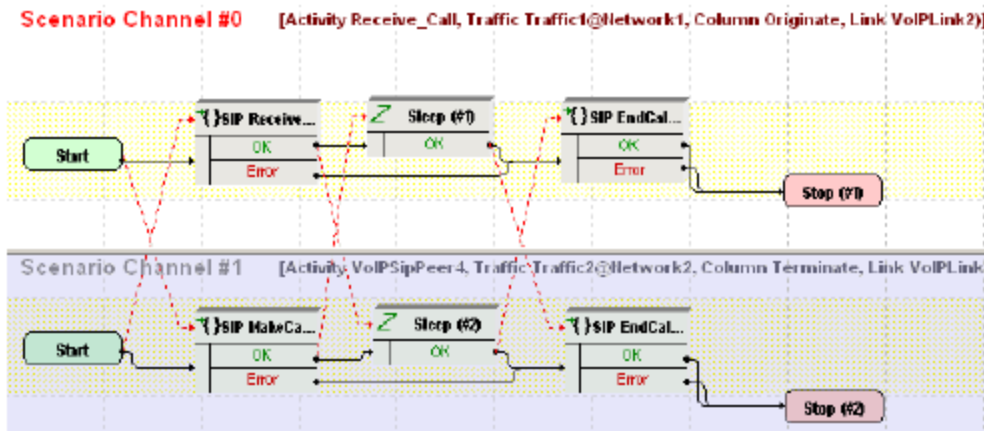
The VoIPSIPCloud1 configuration settings are described in the following table:

Category	Settings
Settings	The SIP cloud emulates three SIP Proxy servers, sip_server#1, sip_server#2, and sip_server#3.
Preview Cloud Traffic	The Make_Call and Registrar activities in the cloud are shown to execute the first and second channels of the SC_004_B2B_SIPv4_T1_MakeCall_from_Cloud_w_Registration test scenario.

SC_005_B2B_SIPv4_T1_ReceiveCall_by_Cloud

This test replicating the [preceding first topology](#) comprises a VoIPSIP Peer and a VoIPSIPCloud activity on the call terminating side, and a VoIPSIPPeer activity on the originating side. The DUT shown in the [preceding first topology](#) is actually not present in the test configuration, the test being run in Back-to-Back mode.

The SIP UAs emulated by Make_Call establish signaling only calls with the endpoints emulated by Receive_Call and configured into the SIP cloud.



The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_005_B2B_SIPv4_T1_ReceiveCall_by_Cloud test scenario comprises a basic call originating functions sequence on the second channel, and a call terminating sequence on the first channel.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The Receive_Call activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page. The SIP server you are running this test against needs specified in the Use external server area, with an outbound proxy functionality selected.
Cloud SIP Settings	This activity is not configured as part of any SIP cloud.

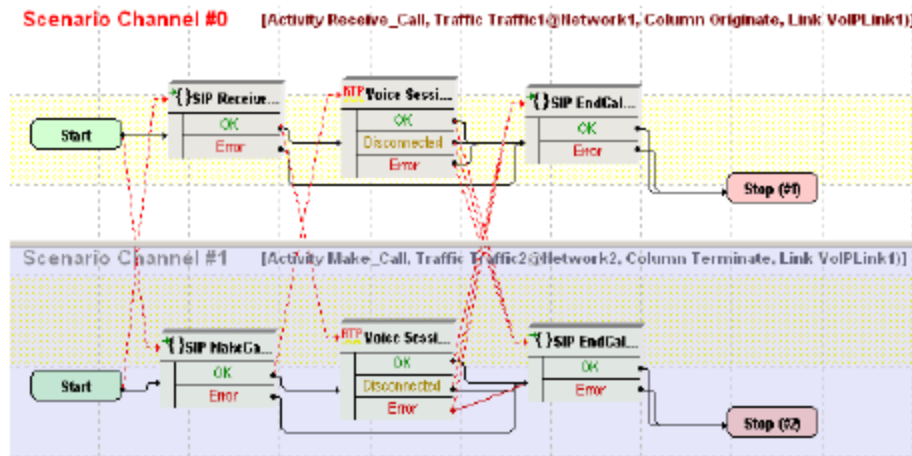
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity. Receive_Call is assigned to VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with **Use Server** (servers are included in the initial messages path) and **Keep in Route** (servers remain in the subsequent messages path) options selected. No overriding dispatching rules are defined.

SC_006_B2B_SIPv4_T1_ReceiveCall_by_Cloud_RTP_n_to_n

This test replicating the [preceding first topology](#) comprises a VoIPSIP Peer and a VoIPSIPCloud activity on the call terminating side, and a VoIPSIPPeer activity on the originating side. The DUT shown in the [preceding first topology](#) is actually not present in the test configuration, the test being run in Back-to-Back mode.


The SIP UAs emulated by Make_Call establish media calls with the endpoints emulated by Receive_Call and configured into the SIP cloud.



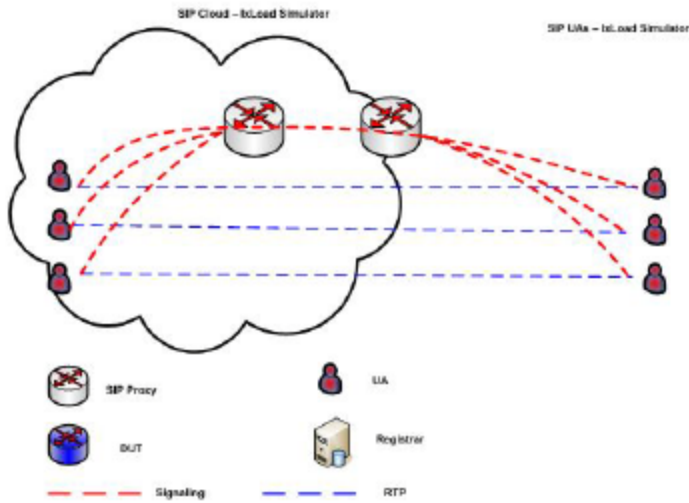
The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_006_B2B_SIPv4_T1_ReceiveCall_by_Cloud_RTP_n_to_n test scenario comprises a basic call originating functions sequence on the first channel, and a call terminating sequence on the second channel. After call establishment, media is exchanged by using the VoiceSession function.

Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) <p>For each media channel, a unique (IP, port) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • UDP port: Use same value
Dial Plan	The Receive_Call activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page. The SIP server you are running this test against needs specified in the Use external server area, with outbound proxy functionality configured.
Cloud SIP Settings	This activity is not configured as part of any SIP cloud.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is selected. The RTP port uses the same 10000 value for all channels.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

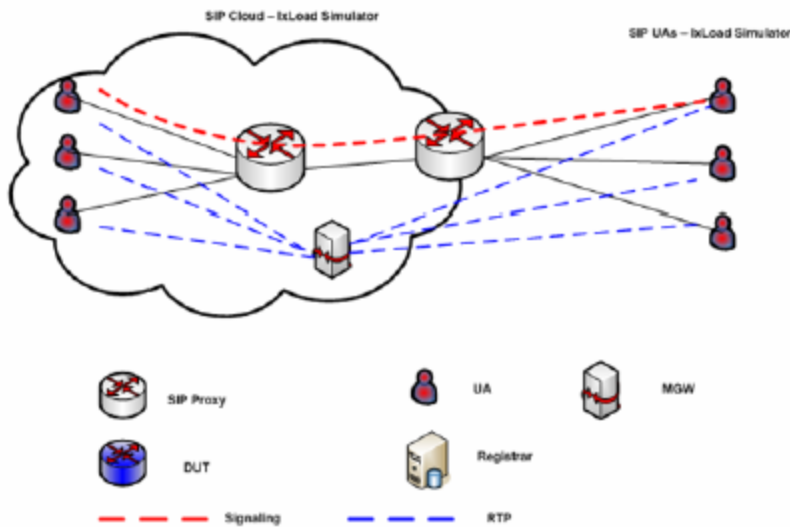
 **Note:** Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity. Receive_Call is assigned to VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with **Use Server** (servers are included in the initial messages path) and **Keep in Route** (servers remain in the subsequent messages path) options selected. No overriding dispatching rules are defined.

Because both originating and terminating side media channels are configured by using multiple IP addresses, the test uses an n-to-n media configuration as shown in the following image:



SC_007_B2B_SIPv4_T1_Receive_Call_by_Cloud_RTP_n_to_1

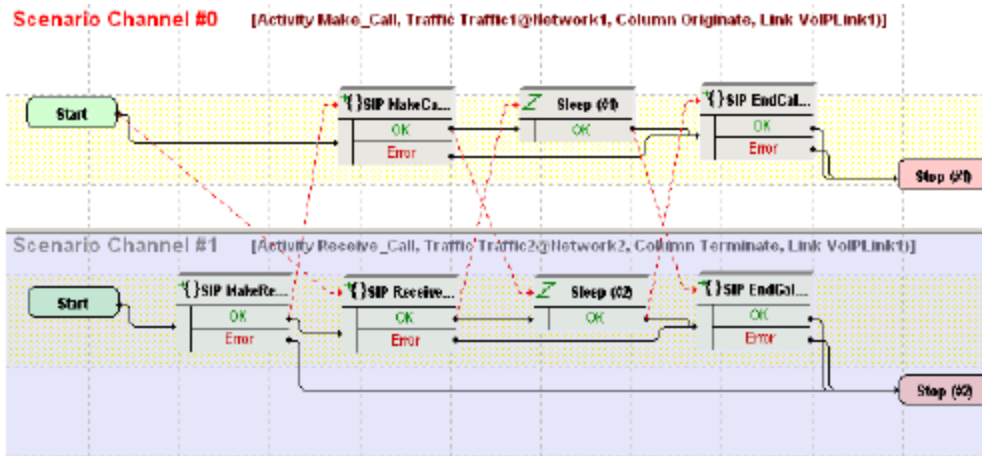
This test is similar to the previous SC_006_B2B_SIPv4_T1_Receive_Call_by_Cloud_RTP_n_to_n test, except for the RTP channels configuration on the Receive_Call activity, which uses the same IP address (different from the signaling IP address) and consecutive UDP ports, resulting in a media configuration as shown in the following image:



SC_008_DUT_SIPv4_T1_MakeCall_from_Cloud_w_Reg_to_DUT

This test replicating the [preceding first topology](#) comprises a VoIPSIPPeer and a VoIPSIPCloud activity on the call terminating side, and a VoIPSIPPeer activity on the originating side.

After registering with a SIP Registrar server (DUT), the SIP UAs emulated by Receive_Call terminate signaling only calls originated by the Make_Call-emulated endpoints.



The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_008_DUT_SIPv4_T1_MakeCall_from_Cloud_w_Reg_to_DUT test scenario comprises a basic call originating functions sequence on the first channel, and a call terminating sequence on the second channel. Before accepting incoming calls, endpoints executing the second channel functions flow first register with a Registrar server.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The Receive_Call activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page. The SIP server (DUT) you are running this test against needs specified in the Use external server area, with outbound proxy functionality selected.
Cloud SIP Settings	This activity is configured into the VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with Use Server (servers are included in the initial messages path) and Keep in Route (servers remain in the subsequent messages path) options selected. No overriding dispatching rules are defined.
Codec Settings	The default codec settings are used.

RTP Settings	Because endpoints establish signaling only calls, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity. The SIP server specified in the Use external server area has outbound proxy and registrar functionality selected.

SC_009_DUT_SIPv4_T1_MakeCall_from_Cloud_w_Reg_to_DUT_RTP

This test is similar to the previous SC_008_B2B_SIPv4_T1_MakeCall_from_Cloud_with_Reg_to_DUT test, except that media is also exchanged on the established calls.

Media endpoints on the originating activity are configured by using a single IP address and consecutive UDP ports. Media endpoints on the terminating endpoints are configured using consecutive IPs and the same port.

SC_010_B2B_SIPv4_T1_MakeCall_ReceiveCall_IM_from_Cloud_RTP

This test replicating the [preceding first topology](#) comprises on one side a VoIPSIPCloud that has three associated VoIPSIPPeers, and three VoIPSIPPeers on the other side. The DUT shown in the [preceding first topology](#) is actually not present in the test configuration, the test being run in Back-to-Back mode.

Make_Call_UDP- and Receive_Call_TCP-emulated endpoints establish and terminate basic signaling and media calls with Receive_Call_UDP- and Make_Call_TCP-emulated endpoints, respectively.


IM_Send_Receive- and IM_Receive_Send- emulated endpoints exchange instant messages both ways.

Each pair of traffic source and destination are configured into an ActivityLink having a corresponding two channel test scenario.

The Make_Call_UDP configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_010_B2B_SIPv4_T1_MakeCall_from_Cloud_RTP tst test scenario comprises a basic call originating sequence on the first channel, and a call terminating sequence on the second channel.


Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) <p>For each media channel, a unique (IP, port) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • UDP port: Use same value
Dial Plan	The Receive_Call activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page.
Cloud SIP Settings	This activity is assigned to the VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with Use Server (servers are included in the initial messages path) and Keep in Route (servers remain in the subsequent messages path) options selected. No overriding dispatching rules are defined.
Codec Settings	The default codec settings are used.
RTP Settings	Because endpoints establish signaling and media calls, the Enable media on this activity option is selected. The RTP port is configured to the same 10000 value for all channels.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** Receive_Call_UDP uses the same settings as Make_Call_UDP, except for the **Dial Plan** page, which does not need to specify a destination activity. In addition, Receive_Call_UDP is not assigned to any cloud.

The Make_Call_TCP configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_010_B2B_SIPv4_T1_ReceiveCall_by_Cloud_RTP test scenario comprises a basic call originating sequence on the first channel, and a call terminating sequence on the second channel.


Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) <p>For each media channel, a unique (IP, port) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • UDP port: Use same value
Dial Plan	The Receive_Call_TCP activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page. The Override transport option is enabled and TCP transport is selected.
Cloud SIP Settings	This activity is not assigned to any cloud.
Codec Settings	The default codec settings are used.
RTP Settings	Because endpoints establish signaling and media calls, the Enable media on this activity option is selected. The RTP port is configured to the same 10000 value for all channels.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** Receive_Call_TCP uses the same settings as Make_Call_TCP, except for the **Dial Plan** page, which does not need to specify a destination activity. In addition, Receive_Call_UDP is assigned to VoIPSIPCloud1, which emulates two Proxy servers, both configured with **Use Server** (servers are included in the initial messages path) and **Keep in Route** (servers remain in the subsequent messages path) options selected.

The IM_Send_Receive configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_010_B2B_SIPv4_T1_Instant_Messaging test scenario comprises a basic instant messaging send and receive sequence on the first channel, and a receive and send sequence on the second channel.

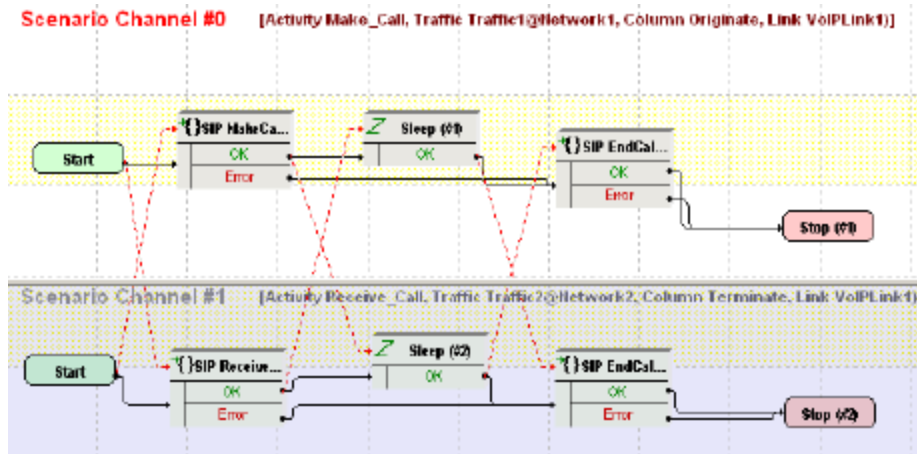
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The IM_Receive_Send activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port = Use same value setting from the Execution Settings page.
Cloud SIP Settings	This activity is assigned to VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with Use Server (servers are included in the initial messages path) and Keep in Route (servers remain in the subsequent messages path) options selected. No overriding dispatching rules are defined.
Codec Settings	The default codec settings are used.
RTP Settings	Because endpoints perform only instant messaging procedures, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** IM_Receive_Send uses the same settings as IM_Send_Receive, except for the **Dial Plan** page, which specifies IM_Send_Receive as destination activity. In addition, IM_Receive_Send is not assigned to any cloud.

SC_011_B2B_SIPv4_T2_BasicCall_between_two_Clouds

This test replicating the [preceding second topology](#) comprises a VoIPSIPPeer assigned to a VoIPSIPCloud both on the call originating and the call terminating side. The DUT shown in the [preceding second topology](#) is actually not present in the test configuration, the test being run in Back-to-Back mode.

The SIP UAs emulated by Receive_Call terminate signaling-only calls originated by the Make_Call-emulated endpoints.



The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_011_B2B_SIPv4_T2_BasicCall_between_two_Clouds test scenario comprises a basic call originating functions sequence on the first channel, and a call terminating sequence on the second channel.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The Receive_Call activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port = Use same value setting from the Execution Settings page.
Cloud SIP Settings	This activity is assigned to VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with Use Server (servers are included in the initial messages path) and Keep in Route (servers remain in the subsequent messages path) options selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because endpoints establish signaling only calls, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity.

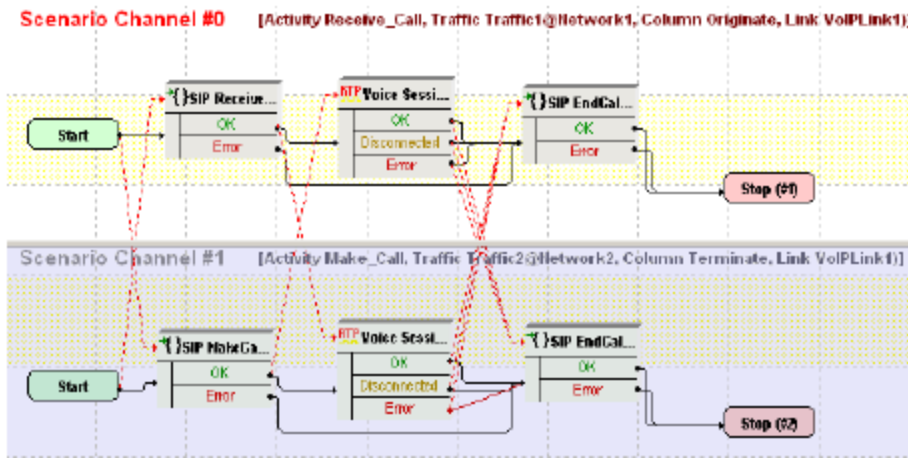
SC_012_B2B_SIPv4_T2_BasicCall_between_two_Clouds_no_Routes

This test is similar to the previous C_011_B2B_SIPv4_T2_BasicCall_between_two_Clouds one, except that no SIP message routing functionality is configured at activity level (the **Keep in route** option in Cloud SIP Settings page is not selected).

SC_013_B2B_SIPv4_T2_BasicCall_between_two_Clouds_RTP_n_to_n

This test replicating the [preceding second topology](#) comprises a VoIPSIPPeer assigned to a VoIPSIPCloud both on the call originating and the call terminating side. The DUT shown in the [preceding second topology](#) is actually not present in the test configuration, the test being run in Back-to-Back mode.


The SIP UAs emulated by Receive_Call terminate signaling and media calls originated by the Make_Call-emulated endpoints. After calls are established, media traffic is exchanged directly between endpoints.



The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_013_B2B_SIPv4_T2_BasicCall_between_two_Clouds_RTP_n_to_n test scenario comprises a basic call originating functions sequence on the first channel, and a call terminating sequence on the second channel.

Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) <p>For each media channel, a unique (IP, port) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • UDP port: Use same value
Dial Plan	The Receive_Call activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page.
Cloud SIP Settings	This activity is assigned to VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with Use Server (servers are included in the initial messages path) and Keep in Route (servers remain in the subsequent messages path) options selected. No overriding dispatching rules are defined.
Codec Settings	The default codec settings are used.
RTP Settings	Because endpoints establish signaling and media calls, the Enable media on this activity option is selected. The RTP port is configured to the same 10000 value for all channels.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity.

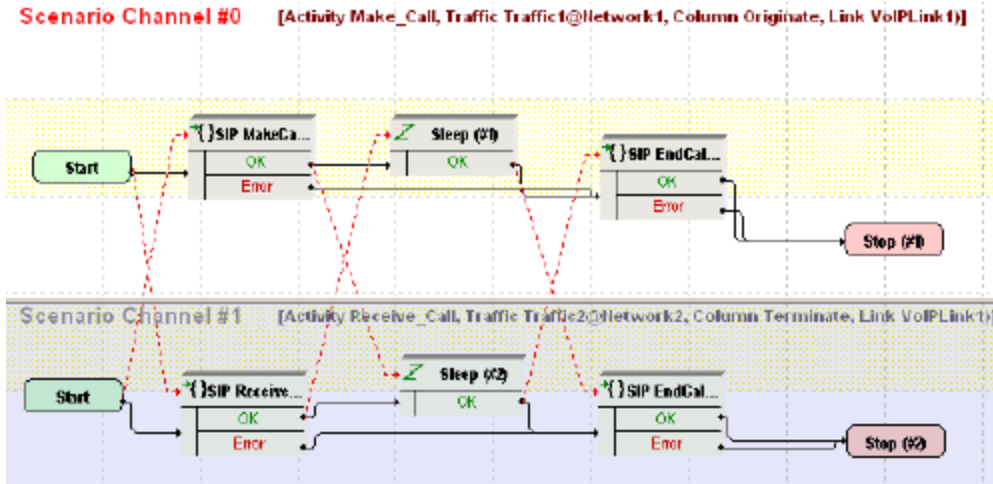
SC_014_B2B_SIPv4_T2_BasicCall_between_two_Clouds_RTP_1_to_1

This test is similar to the previous SC_013_B2B_SIPv4_T2_BasicCall_between_two_Clouds_RTP_n_to_n one, except that media is exchanged between endpoints by sharing a single IP address (different than the signaling ones) and consecutive values for the UDP ports.

SC_015_DUT_SIPv4_T2_BasicCall_between_two_Clouds

This test replicating the [preceding second topology](#) comprises a VoIPSIPPeer assigned to a VoIPSIPCloud both on the call originating and the call terminating side.

The SIP UAs emulated by Receive_Call terminate signaling only calls originated by the Make_Call-emulated endpoints.



The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_015_DUT_SIPv4_T2_BasicCall_between_two_Clouds test scenario comprises a basic call originating sequence on the first channel, and a call terminating sequence on the second channel.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The Receive_Call activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port = Use same value setting from the Execution Settings page. The SIP server (DUT) you are running this test against needs specified in the Use external server area, with outbound proxy functionality selected.
Cloud SIP Settings	This activity is assigned to VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with Use Server (servers are included in the initial messages path) and Keep in Route (servers remain in the subsequent messages path) options selected. No overriding dispatching rules are defined.
Codec Settings	The default codec settings are used.

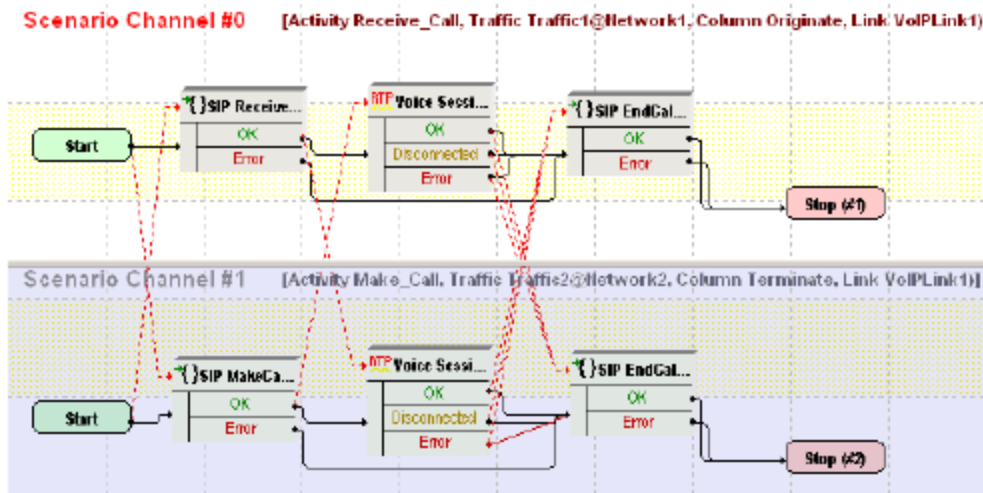
RTP Settings	Because endpoints establish signaling and media calls, the Enable media on this activity option is selected. The RTP port is configured to the same 10000 value for all channels.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity.

SC_016_DUT_SIPv4_T2_BasicCall_between_two_Clouds_RTP_n_to_n

This test replicating the [preceding second topology](#) comprises a VoIPSIPPeer assigned to a VoIPSIPCloud both on the call originating and the call terminating side.


The SIP UAs emulated by Receive_Call terminate signaling and media calls originated by the Make_Call-emulated endpoints.



The Make_Call configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_016_DUT_SIPv4_T2_BasicCall_between_two_Clouds_RTP_n_to_n test scenario comprises a basic call originating sequence on the first channel, and a call terminating sequence on the second channel.

Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) <p>For each media channel, a unique (IP, port) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • UDP port: Use same value
Dial Plan	The Receive_Call activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port = Use same value setting from the Execution Settings page. The SIP server (DUT) you are running this test against needs specified in the Use external server area, with outbound proxy functionality selected.
Cloud SIP Settings	This activity is assigned to VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with Use Server (servers are included in the initial messages path) and Keep in Route (servers remain in the subsequent messages path) options selected. No overriding dispatching rules are defined.
Codec Settings	The default codec settings are used.
RTP Settings	Because endpoints establish signaling and media calls, the Enable media on this activity option is selected. The RTP port is configured to the same 10000 value for all channels.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity.

SC_017_DUT_SIPv4_T2_BasicCall_between_two_Clouds_RTP_1_to_1

This test is similar to the previous SC_017_DUT_SIPv4_T2_BasicCall_between_two_Clouds_RTP_n_to_n one, except that media is exchanged between endpoints sharing a single IP address (different than the signaling ones) and consecutive values for the UDP ports.

SC_018_B2B_SIPv4_T2_MakeCall_ReceiveCall_IM_w_RTP_two_Clouds

This test replicating the [preceding second topology](#) comprises on each side a VoIPSIPCloud having three associated VoIPSIP Peers. The DUT shown in the preceding second topology is actually not present in the test configuration, the test being run in Back-to-Back mode.


Make_Call_UDP and Receive_Call_TCP emulated endpoints on VoIPSIPCloud1 establish and terminate basic signaling and media calls with Receive_Call_UDP and Make_Call_TCP endpoints on VoIPSIPCloud2, respectively.

IM_Send_Receive emulated endpoints on VoIPSIPCloud1 and IM_Receive_Send on VoIPSIPCloud2 exchange instant messages both ways.

Each pair of traffic source and destination are configured into an ActivityLink having a corresponding test scenario.


The Make_Call_UDP configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_018_B2B_SIPv4_T2_MakeCall_w_RTP_two_Clouds tst test scenario comprises a basic call originating sequence on the first channel, and a call terminating sequence on the second channel.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) For each media channel, a unique (IP, port) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • UDP port: Use same value
Dial Plan	The Receive_Call activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page.
Cloud SIP Settings	This activity is assigned to VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with Use Server (servers are included in the initial messages path) and Keep in Route (servers remain in the subsequent messages path) options selected. No overriding dispatching rules are defined.
Codec Settings	The default codec settings are used.
RTP Settings	Because endpoints establish signaling and media calls, the Enable media on this activity option is selected. The RTP port is configured to the same 10000 value for all channels.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** Receive_Call uses the same settings as Make_Call, except for the **Dial Plan** page, which does not need to specify a destination activity.


The Make_Call_TCP configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_018_B2B_SIPv4_T2_ReceiveCall_w_RTP_two_Clouds tst test scenario comprises a basic call originating sequence on the first channel, and a call terminating sequence on the second channel.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port) For each media channel, a unique (IP, port) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • UDP port: Use same value
Dial Plan	The Receive_Call_TCP activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page. The Override transport option is enabled and TCP transport is selected.
Cloud SIP Settings	This activity is assigned to VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with Use Server (servers are included in the initial messages path) and Keep in Route (servers remain in the subsequent messages path) options selected. No overriding dispatching rules are defined.
Codec Settings	The default codec settings are used.
RTP Settings	Because endpoints establish signaling and media calls, the Enable media on this activity option is selected. The RTP port is configured to the same 10000 value for all channels.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** Receive_Call_TCP uses the same settings as Make_Call_TCP, except for the **Dial Plan** page, which does not need to specify a destination activity.

The IM_Send_Receive configuration settings are described in the following table:

Category	Settings
Scenario Editor	The SC_SC_019_B2B_SIPv4_T2_Instant_Messaging_two_Clouds test scenario comprises a basic instant messaging send and receive sequence on the first channel, and a receive and send sequence on the second channel.
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The IM_Receive_Send activity is configured as destination activity.
SIP Settings	The SIP port has the default setting [5060-]; only the first value of the series is used because of the TCP/UDP/TLS port =Use same value setting from the Execution Settings page.
Cloud SIP Settings	This activity is assigned to VoIPSIPCloud1, which emulates two SIP Proxy servers, both configured with Use Server (servers are included in the initial messages path) and Keep in Route (servers remain in the subsequent messages path) options selected. No overriding dispatching rules are defined.
Codec Settings	The default codec settings are used.
RTP Settings	Because endpoints perform only instant messaging procedures, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** IM_Receive_Send uses the same settings as IM_Send_Receive, except for the **Dial Plan** page, which does not need to specify a destination activity.

SC_019_B2B_SIPv6_T1_MakeCall_from_Cloud

This test is similar to [SC_001_B2B_SIPv4_T1_MakeCall_from_Cloud](#), except that the SIP endpoints are configured using IPv6 instead of IPv4 settings.

SC_020_B2B_SIPv6_T1_MakeCall_from_Cloud_RTP_1_to_n

This test is similar to [SC_003_B2B_SIPv4_T1_MakeCall_from_Cloud_RTP_1_to_n](#), except that the SIP endpoints are configured using IPv6 instead of IPv4 settings.

SC_021_B2B_SIPv6_T1_ReceiveCall_by_Cloud

This test is similar to [SC_005_B2B_SIPv4_T1_ReceiveCall_by_Cloud](#), except that the SIP endpoints are configured using IPv6 instead of IPv4 settings.

SC_022_B2B_SIPv6_T2_BasicCall_between_two_Clouds

This test is similar to [SC_011_B2B_SIPv4_T2_BasicCall_between_two_Clouds](#), except that the SIP endpoints are configured using IPv6 instead of IPv4 settings.

SC_023_B2B_SIPv6_T2_BasicCall_between_two_Clouds_no_Routes

This test is similar to [SC_012_B2B_SIPv4_T2_BasicCall_between_two_Clouds_no_Routes](#), except that the SIP endpoints are configured using IPv6 instead of IPv4 settings.

SC_024_B2B_SIPv6_T2_BasicCall_between_two_Clouds_RTP_1_to_1

This test is similar to [SC_014_B2B_SIPv4_T2_BasicCall_between_two_Clouds_RTP_1_to_1](#), except that the SIP endpoints are configured using IPv6 instead of IPv4 settings.

SC_025_B2B_SIPv4_T1_MakeCall_from_Cloud_SRTP_1_to_n

This test is similar to [SC_003_B2B_SIPv4_T1_MakeCall_from_Cloud_RTP_1_to_n](#), except that the media traffic exchanged between endpoint is encrypted by using SRTP.

SC_026_B2B_SIPv4_T2_BasicCall_between_two_Clouds_SRTP_1_to_1

This test is similar to [SC_014_B2B_SIPv4_T2_BasicCall_between_two_Clouds_RTP_1_to_1](#), except that the media traffic exchanged between endpoint is encrypted by using SRTP.

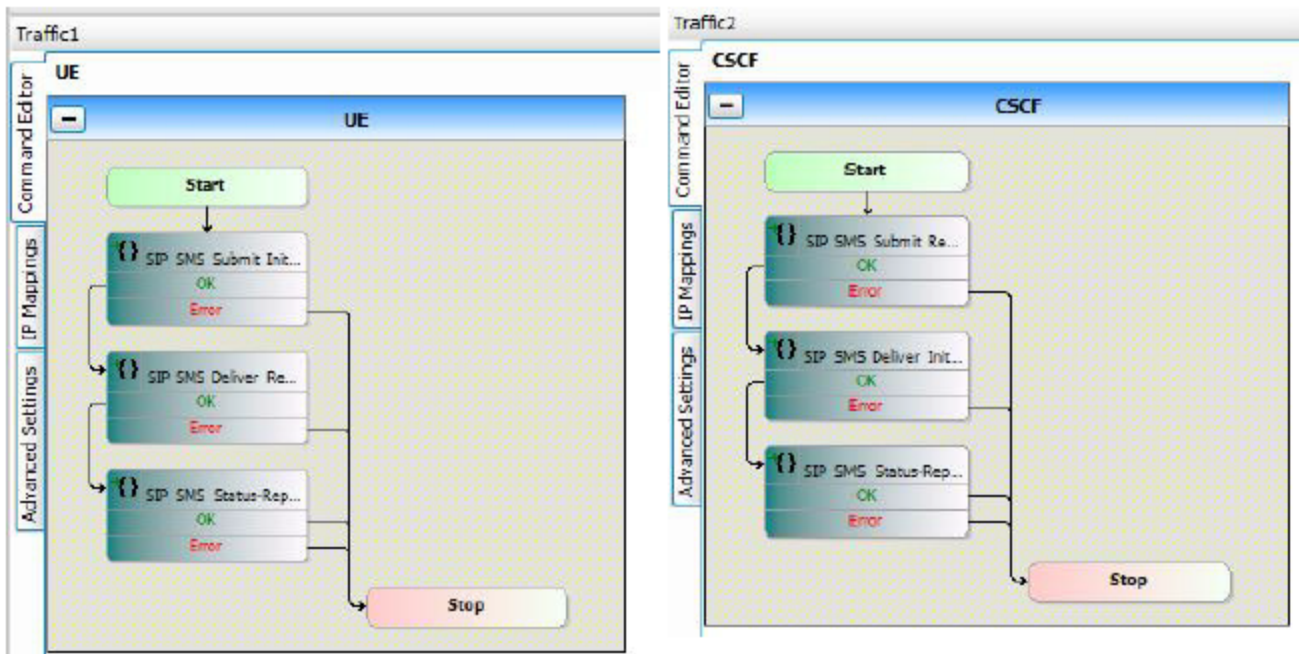
VS_SMS_001_B2B_SIPv4 UE vs. CSCF Complete SMS Flow

This test runs in Back-to-Back mode and comprises of two VoIPSIPPeer activities, each one includes Submit and Deliver and Status-Report procedures. This is a basic SMS flow, where the sender of the SMS corresponds with the receiver.

First activity plays the role of UE and the second represents CSCF. UE sends the SMS Submit message to CSCF, CSCF receives the Submit message, send Deliver to the same UE and then send Submit-Report and because of checked Use Status-Report Indicator.

Category	Settings
Scenario	VS_SMS_001_B2B_SIPv4 UE vs. CSCF Complete SMS Flow comprises two channels: Ch#0: SIP SMS Submit Initiate, SIP SMS Deliver Receive, SIP SMS Status-Report Receive Ch#1: SIP SMS Submit Receive, SIP SMS Deliver Initiate, SIP SMS Status-Report Initiate
Execution Settings	For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	The UE activity and port 5060 are configured as destination for CSCF activity. The CSCF activity and port 5060 are configured as destination for UE activity.

SIP Settings	The SIP port has the 5060 default setting for all channels.
SMS	Use Status Report Request (SM Originator) is set for both UE and CSCF activities. File name sms_user_data.ixsms is added to be sent.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.



VS_SMS_002_DUT_SIPv4 UE vs. UE End-to-end SMS Flow

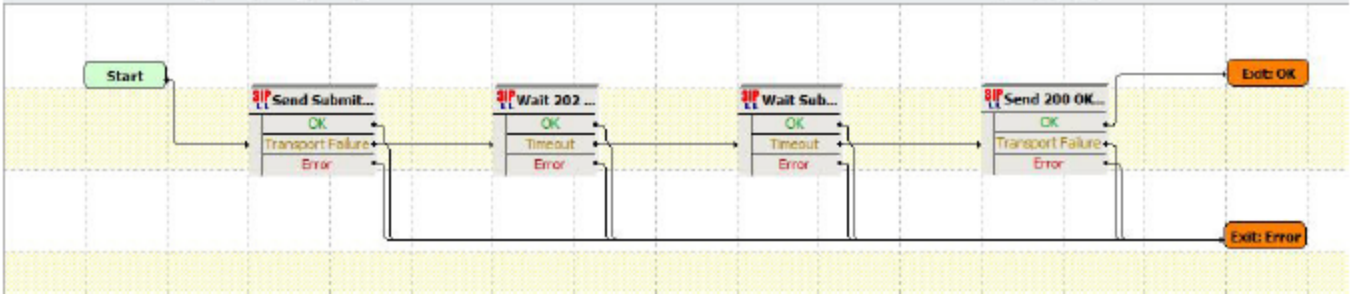
This test runs against DUT (SUT) and comprises of two VoIPSIPPeer activities, first includes the sender of SM (Sender_UE) and the second represents the receiver of SM (Recipient_UE).

In the scenario channels, there are used two procedures: SMS Submit Initiate for Sender_UE, and SMS Deliver Receiver for Recipient_UE.

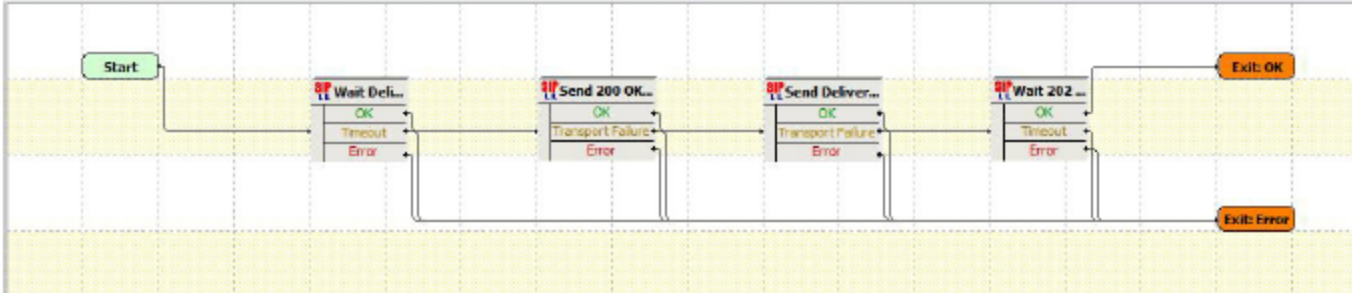
Category	Settings
Scenario	VS_SMS_002_DUT_SIPv4 UE vs. UE End-to-end SMS flow comprises two channels: Ch#0: SIP SMS Submit Initiate Ch#1: SIP SMS Deliver Receive

Execution Settings	<p>For each signaling channel, a unique (IP, port, phone) tuple is generated by using the following Channel Mapping settings:</p> <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • TCP/UDP/TLS port: Use same value • Phone: Use consecutive values (per port)
Dial Plan	<p>The Sender_UE activity and port 5060 are configured as destination for Recipient_UE activity. The Recipient_UE activity and port 5060 are configured as destination for Sender_UE activity.</p>
SIP Settings	<p>The SIP port has the 5060 default setting for all channels.</p>
SMS	<p>File name sms_user_data.ixsms is added to be sent.</p>
Other Settings	<p>The IP version preference is set to IPv4, and no scenario variables need to be initialized.</p>

Scenario Editor: VS_SMS_002_DUT_SIPv4 UE vs. UE End-to-end SMS flow >> SIP SMS Submit Initiate (ID1) (lib) .



Scenario Editor: VS_SMS_002_DUT_SIPv4 UE vs. UE End-to-end SMS flow >> SIP SMS Deliver Receive (ID2) (lib) .



Skinny Procedures, Sample Test Configurations, and Test Scenarios

To get you started more easily with the Skinny protocol testing, a number of predefined Skinny procedures and sample test configuration building up on existing script functions were developed. This section provides a description of these predefined IxLoad Voice Plug-in Skinny procedures, available sample test configurations (RXFs) and their associated test scenarios.

Note: Skinny procedures and sample test configuration are available only when installing the Cisco SCCP library, as described in Appendix A of the Getting Started with IxLoad manual.

Note: For a complete description of the available Skinny test library functions, see [VoIP Skinny Functions Library](#).

Sample Test Configurations Naming Convention

The available test configurations adhere to a naming convention that eases identification of a test composition. Acronyms used for denominating configurations and their meaning are listed in the following table:

Field	Description
Protocol	Skinny, Mixed (SIP and Skinny)
nnn	Sequential number (001-999)
Phone type	Emulated IP phone type (7902, 7960)
Test type	Signaling Only (SO), Signaling and Media (SM)
Network type	User Side (US), Network Side (NS), User Side and Network Side (USNS)
Objective value	Numerical value
Objective type	Channels (Chs), BHCA
IP version	IPv4, IPv6
IP Type	DHCP, Static
Short Description	The name of the sample file

For example, SK_001_7960_SO_US_5000_ChS_IPv4_Static_Basic_Call_10s denotes a Skinny signaling-only test, emulating User Side equipment, having an objective of 5000 channels, by using IPv4 and DHCP assigned addresses, and executing a basic call with a duration of 10 seconds.

Note: Other possible descriptors, such as HL or LL, can also be used to differentiate between library types, such as Low Level (LL) or High Level (HL).

Skinny Predefined Procedures

A procedure is a simple way to encapsulate several script functions into a single function block that can be re-used within a number of scenarios. Based on the functions from the Skinny test library, the Skinny predefined procedures described in the following table (available in the Procedure Library) were developed:

Category	Procedure	Description
High Level	SK_IsOnHook	This procedure retrieves the values of the Skinny scenario variables and uses a VariableTest script function to check if the call state is 'onhook' (\$Sk_CallState=2).
	SK_IsOffHook	This procedure retrieves the values of the Skinny scenario variables and uses a VariableTest script function to check if the call state is 'onhook' (\$Sk_CallState=1).
	SK_IsRingIn	This procedure retrieves the values of the Skinny scenario variables and uses a VariableTest script function to check if the call state is 'onhook' (\$Sk_CallState=4).
	SK_IsRingOut	This procedure retrieves the values of the Skinny scenario variables and uses a VariableTest script function to check if the call state is 'onhook' (\$Sk_CallState=3).
	SK_IsConnected	This procedure retrieves the values of the Skinny scenario variables and uses a VariableTest script function to check if the call state is 'onhook' (\$Sk_CallState=5).
	SK_IsHold	This procedure retrieves the values of the Skinny scenario variables and uses a VariableTest script function to check if the call state is 'hold' (\$Sk_CallState=8).
	SK_REG_7902	This procedure registers a 7902 type phone with a CallManager.
	SK_REG_7910	This procedure registers a 7910 type phone with a CallManager.
	SK_REG_7940	This procedure registers a 7940 type phone with a CallManager.
	SK_REG_7960	This procedure registers a 7960 type phone with a CallManager.
	SK_DeREG	This procedure de-registers a phone.


	SK_Receive and AnswerCall_HL	This procedure, comprising a Wait Call and an Answer Call script function, is the correspondent for the receiving channel of a Make Call function.
Low Level	SK_Empty Message Queue_LL	This procedure empties the Skinny messages queue.
	SK_Make Call_LL	This procedure originates a call by dialing a dial-plan specified number and waiting for a Connected message. It uses Low Level Skinny functions, softkeys and events for initiating a call.
	SK_Receive Call_LL	This procedure uses Low Level Skinny functions, softkeys and events for receiving a call.
	SK_Make Call to BLLL	The procedure originates a call by dialing a Skinny dial-plan specified number and waits for a StartToneMessage with a DtAlertingTone value.
	SK_Hold_LL	This procedure uses Low Level Skinny functions, softkeys and events for putting a call on hold.
	SK_Retrieve_LL	This procedure uses Low Level Skinny functions, softkeys and events for retrieving a call from the hold state.
	SK_End Call Initiate_LL	This procedure uses Low Level Skinny functions, softkeys and events to initiate a call closure.
	SK_End Call Terminate_LL	This procedure uses Low Level Skinny functions, softkeys and events to respond to a call closure.

Skinny Test Configurations

This section provides a brief description of the sample VoIPSkinnyPeer test configuration files that can be installed and used with the IxLoad Voice Plug-in.

Skinny tests fall into either of the following categories:

- [Skinny Signaling Only](#)
- [Skinny Bulk Calls](#)
- [Advanced Call Features](#)
- [Mixed Skinny and SIP - SIP UAs](#)
- [Mixed Skinny and SIP - SIP Trunk](#)

 **Note:** An overview table comprising the most important configured parameters for all Skinny sample test configuration is provided in Appendix E, Skinny Sample Configurations Overview.

Skinny Signaling Only

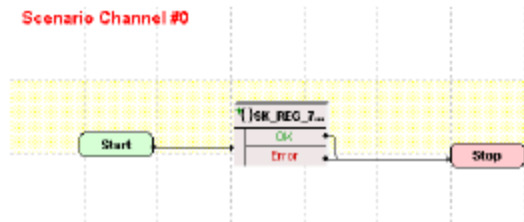
Tests in this category execute Skinny registration and de-registration procedures with a specified Cisco CallManager. Tests comprise a single activity configured to emulate a number of 100 phones that execute the registration and de-registration operations repeatedly during the test sustain time.

SK_001_7902_SO_US_100_ChS_IPv4_Static_Seq_Registration_5_retries

This signaling only test illustrates a phone registration procedure, whereby a number of 7902 type phones emulated by the SK_SEP7902A0000001 activity attempt registration with a specified Cisco CallManager.

This registration procedure is executed repeatedly for the entire sustain time duration.

The underlying one-channel test scenario comprising the functions flow executed by the SK_SEP7902A0000001 emulated phones is shown in the following image:



SK_SEP7902A0000001 configured settings are described in the following table:

Category	Settings
Scenario Editor	The test scenario is completely configured and comprises the predefined Skinny SK_REG_7902 procedure. Within the procedure, phone A on the first scenario channel executes a registration with the CallManager specified in the Skinny Settings page. Registration is attempted successively for five times (the RetriesNo variable is initialized to a value of '5.') If the registration cannot be completed successfully after the fifth attempt, the procedure exits on the Error output.
Execution Settings	The corresponding scenario channel is configured to execute one loop during sustain time and to use one alias/channel.
Dial Plan	The phone registration names are defined by using a SEP7902A00[00001-] sequence generating expression that generates 100 registration name strings. Call and transfer destination need not be configured.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The Sequential registration and the Fail if previously has failed options are selected.
Codec Settings	The codec settings can be left unchanged.

RTP Settings	Because the activity does not perform any RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

SK_002_7960_SO_US_100_ChS_IPv4_Static_Seq_Registration_5_retries

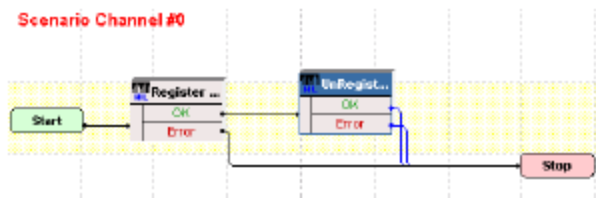
This signaling only test illustrates a phone registration procedure, with the difference that the emulated Skinny phones are of the 7960 type.

SK_005_7902_SO_US_100_ChS_IPv4_Static_Seq_Bulk_Registration_loop

This signaling only test illustrates a phone registration procedure, whereby a number of 7902 type phones emulated by the SK_SEP7902A0000001 activity continuously attempt registration followed by de-registration with the Cisco CallManager. Registration is followed by de-registration.

This registration and de-registration procedure is executed repeatedly for the entire sustain time duration.

The underlying one-channel test scenario is shown in the following image:



SK_SEP7902A0000001 configured settings are described in the following table:

Category	Settings
Scenario Editor	The test scenario is completely configured and comprises the Register Client and Unregister Client script functions.
Execution Settings	The corresponding scenario channel is configured to execute one loop during sustain time and to use one alias/channel.
Dial Plan	The phone registration names are defined by using a SEP7902A00[00001-] sequence generating expression that generates 100 registration name strings. Call and transfer destination need not be configured.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The Sequential registration and the Fail if previously has failed options are selected.
Codec Settings	The codec settings can be left unchanged.

RTP Settings	Because the activity does not perform any RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

SK_006_7960_SO_US_100_ChS_IPv4_Static_Seq_Bulk_Registration_loop

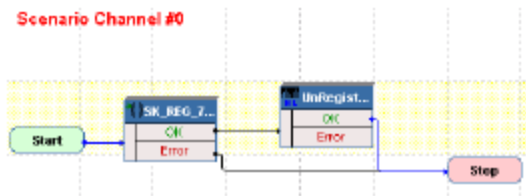
This signaling only test illustrates a phone registration procedure similar to the previous test, with the only difference that the emulated Skinny IP phones are of the 7960 type.

SK_003_7902_SO_US_100_ChS_IPv4_Static_Seq_Bulk_Registration_loop_5_retries

This signaling only test illustrates a phone registration procedure, whereby a number of 7902 type phones emulated by the SK_SEP7902A0000001 activity continuously attempt registration followed by de-registration with the Cisco Call-Manager.

This registration and de-registration procedure executes repeatedly for the entire test sustain time duration.

The underlying one-channel test scenario is shown in the following image:



SK_SEP7902A0000001 configured settings are described in the following table:

Category	Settings
Scenario Editor	The test scenario channel comprises the Sk_REG_7902 procedure followed by a Skinny Unregister Client script function. Using the Sk_REG_7902 procedure, registration is attempted successively for five times (the RetriesNo variable is initialized to a value of '5.'). If the registration cannot be completed successfully after the fifth attempt, the procedure exits on the Error output.
Execution Settings	The corresponding scenario channel is configured to execute one loop during sustain time and to use one alias/channel.
Dial Plan	The phone registration names are defined by using a SEP7902A00[00001-] sequence generating expression that generates 100 registration name strings. Call and transfer destination need not be configured.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The Sequential registration and the Fail if previously has failed options are selected.

Codec Settings	The codec settings can be left unchanged.
RTP Settings	Because the activity does not perform any RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

SK_004_7960_SO_US_100_ChS_IPv4_Static_Seq_Bulk_Registration_loop_5_retries

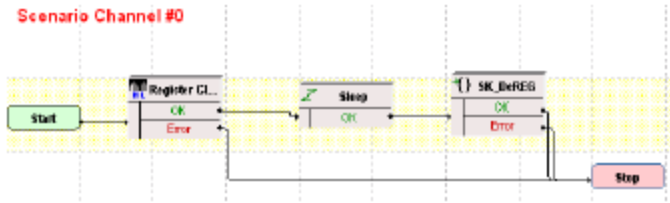
This signaling only test illustrates a phone registration procedure similar to that of the previous test, with the difference that the emulated Skinny phones are of the 7960 type.

SK_007_7902_SO_US_100_ChS_IPv4_Static_Seq_Reg_5s_Sleep_Dereg_5_retries

This signaling only test illustrates a phone registration procedure, whereby a number of 7902 type phones emulated by the SK_SEP7902A0000001 activity attempt de-registration with the Cisco CallManager. After being registered for a five seconds duration, the phones de-register.

This registration and de-registration sequence executes repeatedly for the entire sustain time duration.

The underlying one-channel test scenario is shown in the following image:



SK_SEP7902A0000001 configured settings are described in the following table:

Category	Settings
Scenario Editor	The test scenario channel comprises the Skinny Register Client function, followed by a Sleep function to maintain the registration for a five seconds duration. Phones de-registrations finally occurs by using the Skinny DeREG procedure. Using the DeREG procedure, de-registration is attempted successively for five times (the RetriesNo variable is initialized to a value of `5.`) If the de-registration cannot be completed successfully after the fifth attempt, the procedure exits on the Error output.
Execution Settings	The corresponding scenario channel is configured to execute one loop during sustain time and to use one alias/channel.
Dial Plan	The phone registration names are defined by using a SEP7902A00[00001-] sequence generating expression that generates 100 registration name strings. Call and transfer destination need not configured.

Skinnny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The Sequential registration and the Fail if previously has failed options are selected.
Codec Settings	The codec settings can be left unchanged.
RTP Settings	Because the activity does not perform any RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

SK_008_7960_SO_US_100_ChS_IPv4_Static_Seq_Reg_5s_sleep_Dereg_5_retries

This signaling only test illustrates a phone de-registration procedure similar to that of the previous test, with the difference that the emulated Skinny phones are of the 7960 type.

Skinny Bulk Calls

This test samples group is aimed at stressing the Cisco CallManager with bulk calls, both signaling-only and signaling with media streaming, between emulated Cisco 7902 IP phones.

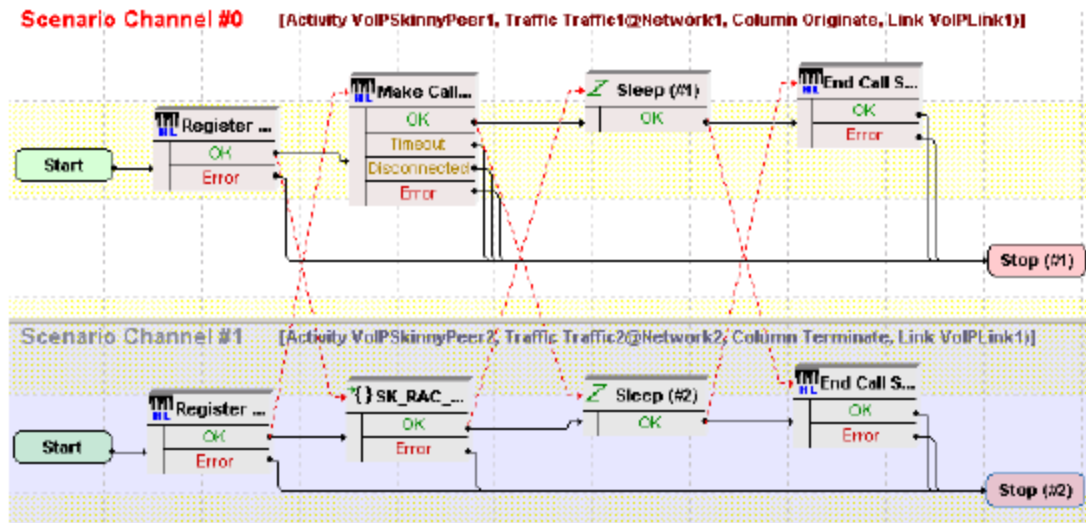
Activities in this test group are configured to emulate large number of phones and to execute repeatedly during the test sustain time. Both signaling only and signaling with media streaming test samples are available.

SK_009_7902_SO_US_5000_ChS_IPv4_Static_Basic_Call_10s

This signaling only test illustrates a basic call procedure, whereby a number of 7902-type Skinny phones emulated by the SK_SEP7902A0000001 activity establish calls with the Skinny phones emulated by the SK_SEP7902B0000001 activity. After establishing a call, the phones remain in an active call state for the duration of the Sleep script function before the call is terminated.

This call procedure is executed repeatedly for the entire test sustain time.

The underlying two-channel test scenario involving Skinny phones A and B is shown in the following image:



Category	Settings
Scenario Editor	The test scenario comprises phone registrations with the Cisco CallManager followed by a call establishment procedure. Registration is performed during the first loop only. Phone A of the first scenario channel uses a Skinny Make Call script function to initiate a call based on the Dial Plan settings. The Sk_Receive AnswerCall_HL procedure used by phone B of the second channel includes the common Skinny Wait Call and Skinny Answer Call script functions. After establishing the call, Sleep function, with a 10 seconds duration, is executed by both phone A and phone B. Eventually, the call is terminated by phone A by using a Skinny End Call procedure.
Execution Settings	The corresponding scenario channel is configured to execute one loop during sustain time and to use one alias/channel.
Dial Plan	The phone registration names are defined by using a SEP7902A00[00001-] sequence generating expression. SK_SEP7902B0000001 is configured as a call destination.
Skiny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The Sequential registration and the Fail if previously has failed options are selected.
Codec Settings	The codec settings are of no importance and can be left unchanged, because no RTP streaming is performed by the test.
RTP Settings	Because the activity does not perform any RTP streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: SK_SEP7902B0000001 uses the same settings as SK_SEP7902A0000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. The registration names of the SK_SEP7902B0000001 - emulated phones are specified by using a SEP7902B00 [00001-] sequence generating expression.

SK_010_7902_SO_US_5000_ChS_IPv4_Static_Basic_Call_3min

This test is similar to the previous one, with the only difference that the call duration is three minutes.

SK_011_7902_SO_US_5000_ChS_IPv4_Static_Basic_Call_30min

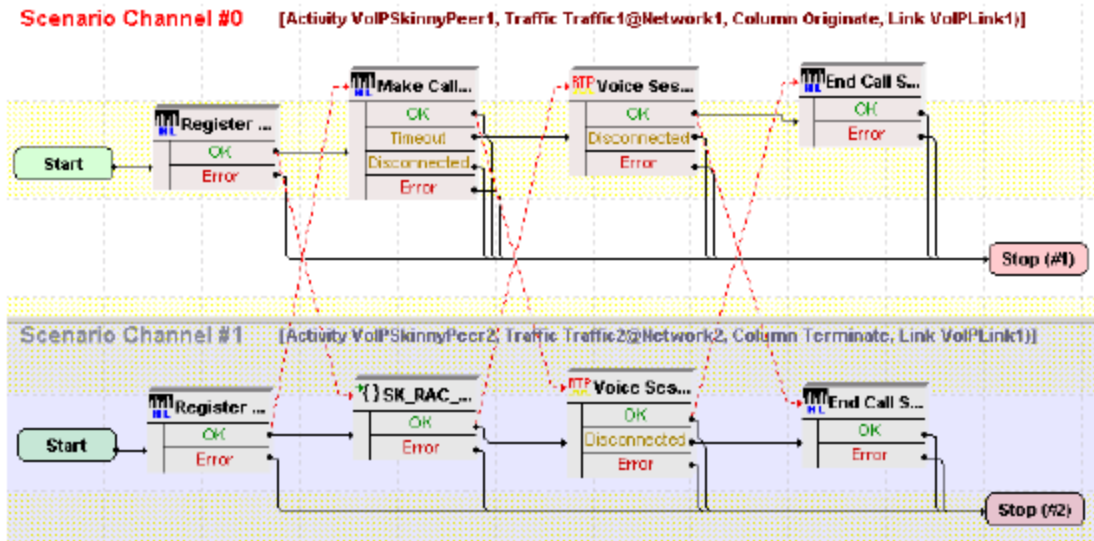
This test is similar to the previous one, with the only difference that the call duration is 30 minutes.

SK_012_7902_SM_US_900_ChS_IPv4_Static_Basic_Call_Voice_10s

This test illustrates a basic call procedure with media streaming, whereby a number of 7902 phones emulated by the SK_SEP7902A0000001 activity establish calls with the Skinny phones emulated by the SK_SEP7902B0000001 activity. After establishing a call, the phones perform bidirectional media streaming before the call is terminated.

This call procedure is repeated for the entire test sustain time.


The underlying two channel test scenario involving Skinny phones A and B is shown in the following image:



SK_SEP7902A0000001 configured settings are described in the following table:

Category	Settings
----------	----------

Scenario Editor	The test scenario channel comprises the phones registration with the CallManager followed by a call establishment procedure. Phone A of the first scenario channel uses a Skinny Make Call script function to initiate a call based on the Dial Plan settings. After establishing the call, media streaming is performed by using Voice Session script functions on both scenario channels. Finally, the call is terminated by phone A by using a Skinny End Call procedure. The Sk_Receive AnswerCall_HL procedure used by phone B on the second channel is a wrapping of the common Skinny Wait Call and Skinny Answer Call script functions.
Execution Settings	The corresponding scenario channel is configured to execute one loop during sustain time and to use one alias/channel.
Dial Plan	The phone registration names are defined by using a SEP7902A00[00001-] sequence generating expression. SK_SEP7902B0000001 is configured as a call destination.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The Sequential registration and the Fail if previously has failed options are selected.
Codec Settings	The codec settings can be left unchanged.
RTP Settings	The Enable media on this activity option is selected for media streaming to be performed between the emulated Skinny phones.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** SK_SEP7902B0000001 uses the same settings as SK_SEP7902A0000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. The registration names of the SK_SEP7902B0000001-emulated phones are specified by using a SEP7902B00 [00001-] sequence generating expression.

SK_013_7902_SM_US_900_ChS_IPv4_Static_Basic_Call_Voice_3min

This test is similar to the previous one, with the only difference that the call duration is three minutes. The longer call duration is obtained by playing the selected

wave file continuously for three minutes (**Advanced Playback Settings** tab of the Voice Session script function).

SK_014_7902_SM_US_900_ChS_IPv4_Static_Basic_Call_Voice_30min

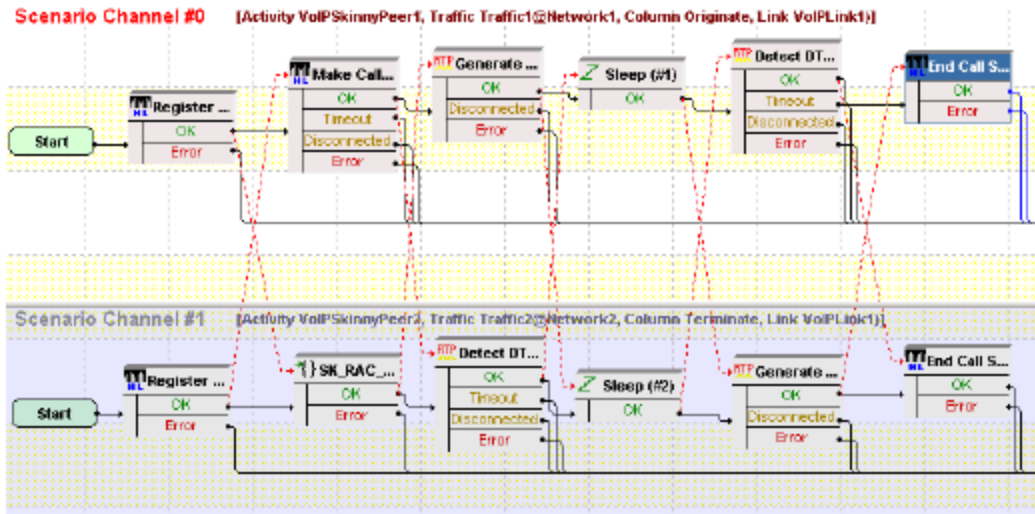
This test is similar to the previous one, with the only difference that the call duration is 30 minutes. The longer call duration is obtained by playing the selected wave file continuously for three minutes (**Advanced Playback Settings** tab of the Voice Session script function).

SK_016_7902_SM_US_300_ChS_IPv4_Static_Basic_Call_DTMFs_inband_3min_7902

This test illustrates a basic call procedure with inband transmission of DTMFs. A number of 7902 phones emulated by the SK_SEP7902A0000001 activity establish calls with the phones emulated by the SK_SEP7902B0000001 activity. After the call is established, the phones use the Generate DTMF/Detect DTMF script functions for bidirectional transmission and detection of DTMFs.

This call procedure is repeated for the entire test sustain time.

The underlying two-channel test scenario involving phones A and B is shown in the following image:



SK_SEP7902A0000001 configured settings are described in the following table:

Category	Settings
Scenario Editor	The test scenario comprises the phones registration followed by a call establishment procedure. Phone A on the first scenario channel uses a Skinny Make Call script function to initiate a call based on the Dial Plan settings. The Sk_Receive AnswerCall_HL procedure used by phone B on the second channel is a wrapping of the common Skinny Wait Call and Skinny Answer Call script functions. After establishing the call, DTMFs are transmitted bidirectionally inband by using Generate DTMF script functions having the inband transmission mode selected. DTMFs are detected by using Detect DTMF functions. Eventually, the call is terminated by phone A by using a Skinny End Call procedure.
Execution Settings	The corresponding scenario channel is configured to execute one loop during sustain time and to use one alias/channel.
Dial Plan	The phone registration names are defined by using a SEP7902A00[00001-] sequence generating expression. SK_SEP7902B0000001 is configured as a call destination.

Skippy Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The Sequential registration and the Fail if previously has failed options are selected.
Codec Settings	The G.711 codec is selected for the DTMFs to be transmitted inband.
RTP Settings	The Enable media on this activity option is selected for the DTMFs to be transmitted inband.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: SK_SEP7902B0000001 uses the same settings as SK_SEP7902A0000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. The registration names of the SK_SEP7902B0000001 - emulated phones are specified by using a SEP7960B00 [00001-] sequence generating expression.

SK_017_7902_SM_US_300_ChS_IPv4_Static_Basic_Call_DTMFs_out-of-band_3min_7902

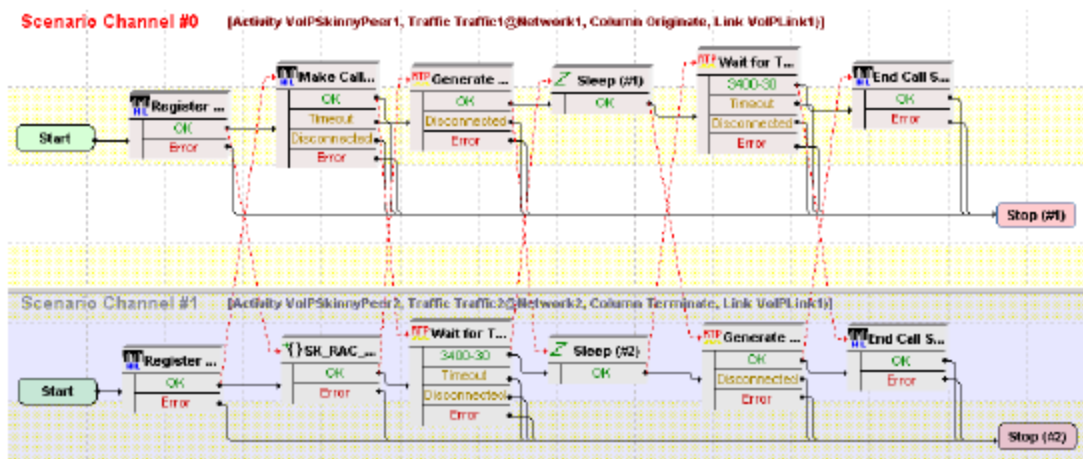
This test is similar to the previous one, with the difference that the DTMFs are transmitted out-of-band by using the RTP 2833 Event payload format in the Generate DTMF script functions.

SK_021_7902_SM_US_300_ChS_IPv4_Static_BasicCall_Tone_highFreq_inband_3min

This test illustrates a basic call procedure with inband transmission and detection of custom tones. A number of 7902 phones emulated by the SK_SEP7902A0000001 activity establish calls with the phones emulated by the SK_SEP7902B0000001 activity. After establishing the call, the phones use the Generate Tone/Wait for Tone script functions for transmitting and detecting custom tones.


This call procedure is repeated for the entire test sustain time.

The underlying two-channel test scenario involving phones A and B is shown in the following image:



SK_SEP7902A0000001 configured settings are described in the following table:

Category	Settings
Scenario Editor	The test scenario comprises the phones registration followed by a call establishment procedure. Phone A on the first scenario channel uses a Skinny Make Call script function to initiate a call based on the Dial Plan settings. The Sk_Receive AnswerCall_HL procedure used by phone B on the second channel is a wrapping of the common Skinny Wait Call and Skinny Answer Call script functions. After establishing the call, DTMFs are transmitted bidirectionally inband by using Generate DTMF script functions having the inband transmission mode selected. DTMFs are detected by using Detect DTMF functions. Eventually, the call is terminated by phone A by using a Skinny End Call procedure.
Execution Settings	The corresponding scenario channel is configured to execute one loop during sustain time and to use one alias/channel.
Dial Plan	The registration names are defined by using a SEP7902A00[00001-] sequence generating expression. SK_SEP7902B0000001 is configured as call destination and no transfer destination needs to be configured.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The Sequential registration and the Fail if previously has failed options are selected.
Codec Settings	The G.711 codec is selected for the tone to be transmitted inband.
RTP Settings	The Enable media on this activity option is selected for the tone to be transmitted inband.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** SK_SEP7902B0000001 uses the same settings as SK_SEP7902A0000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. The registration names of the SK_SEP7902B0000001-emulated phones are specified by using a SEP7902B00[00001-] sequence generating expression.

SK_020_7902_SM_US_300_ChS_IPv4_Static_BasicCall_Tone_medFreq_inband_3min

The test is the same as the previous one, with the difference that a different custom tone is transmitted by the Generate Tone functions.

SK_019_7902_SM_US_300_ChS_IPv4_Static_BasicCall_Tone_lowFreq_inband_3min

The test is the same as the previous one, with the difference that a different custom tone is transmitted by the Generate Tone functions.

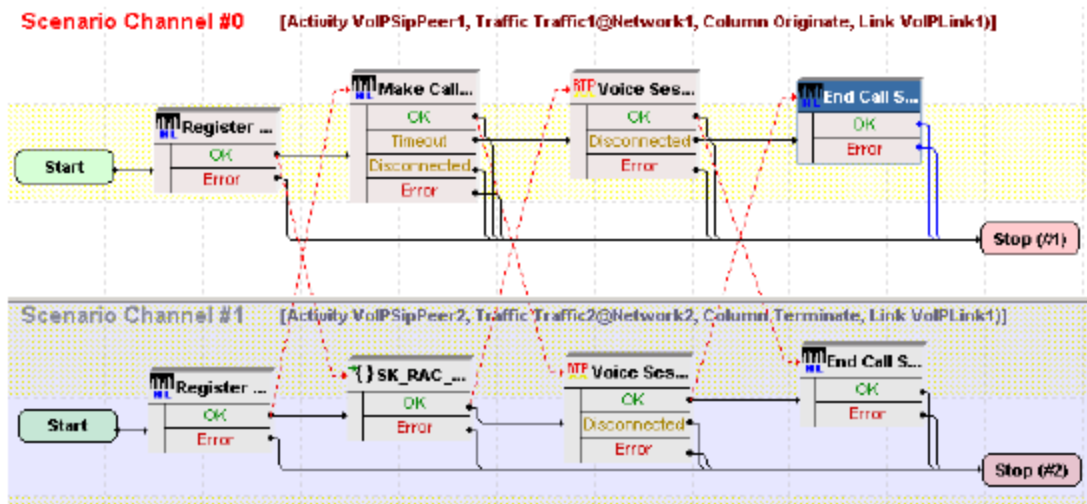
SK_015_7902_SM_US_10K_BHCA_IPv4_Static_Basic_Call_Voice_1min

This test illustrates a basic call procedure with media streaming, having a configured BHCA test objective of 10000 calls/hour. A number of 7902 type phones emulated by the SK_SEP7902A0000001 activity establish calls with the phones emulated by the SK_SEP7902B0000001 activity. After establishing the call, the phones use the Voice Session functions for bidirectional media streaming.

Note: The BHCA objective of 10000 calls/hour is configured with a talk time of 40 seconds, slightly longer than the duration of the wave file played by the Voice Session functions on each scenario channel.

This call procedure is executed repeatedly for the entire test sustain time.

The underlying two-channel test scenario involving phones A and B is shown in the following image:



SK_SEP7902A0000001 configured settings are described in the following table:

Category	Settings
Scenario Editor	The test scenario comprises the phones registration with the Call Manager by using the Dial Plan settings, followed by a call establishment procedure. Phone A on the first scenario channel uses a Skinny Make Call script function to initiate a call based on the Dial Plan settings. The Sk_Receive AnswerCall_HL procedure used by phone B on the second channel is a wrapping of the common Skinny Wait Call and Skinny Answer Call script functions. The Voice Session functions used on both channels for bidirectional media streaming each play a wave file with a duration of about 33 seconds. Eventually, the call is terminated by phone A by using a Skinny End Call procedure.
Execution Settings	The corresponding scenario channel is configured to execute one loop during sustain time and to use one alias/channel.
Dial Plan	The registration names are defined by using a SEP7902A00[00001-] sequence generating expression. SK_SEP7902B0000001 is configured as call destination and no transfer destination needs to be configured.

Skippy Settings	The Enable signaling on this activity option is selected for the Skippy functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The Sequential registration and the Fail if previously has failed options are selected.
Codec Settings	Default codec settings are used.
RTP Settings	The Enable media on this activity option is selected for media streaming to be performed between the emulated phones.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: SK_SEP7902B0000001 uses the same settings as SK_SEP7902A0000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. The registration names of the SK_SEP7902B0000001 - emulated phones are specified by using a SEP7902B00 [00001-] sequence generating expression.

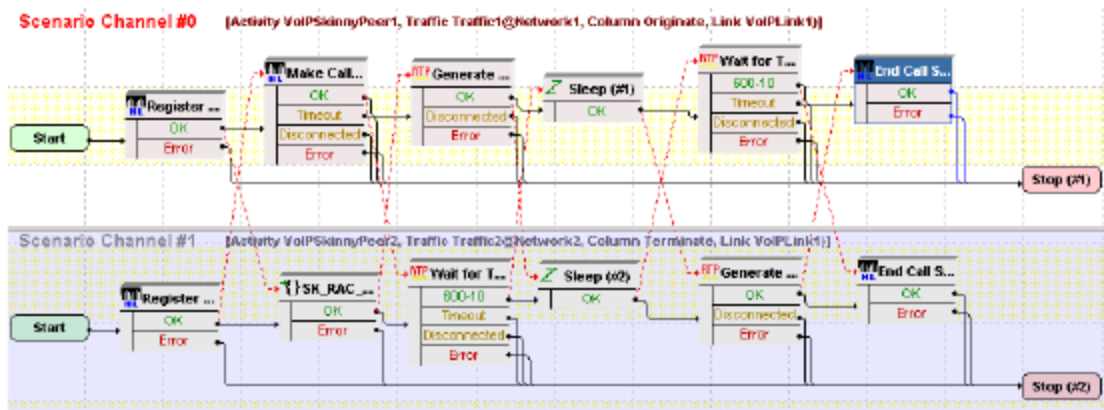
SK_018_7902_SM_US_25K_BHCA_IPv4_Static_Basic_Call_DTMFs_inband_3_min

This test illustrates a basic call procedure with inband transmission of custom tones, having a configured BHCA test objective of 25000 calls/hour. The 7902 phones emulated by the SK_SEP7902A0000001 activity establish calls with the phones emulated by the SK_SEP7902B0000001 activity. After establishing the call, the phones use the Generate Tone/Wait for Tone functions for generating and detecting custom tones.

Note: The BHCA objective of 25000 calls/hour is configured with a talk time of approximately 120 seconds.


This call procedure is repeated for the entire test sustain time.

The underlying two-channel test scenario phones A and B is shown in the following image:



SK_SEP7902A0000001 configured settings are described in the following table:

Category	Settings
Scenario Editor	The test scenario comprises the phones registration followed by a call establishment procedure. After establishing the call, DTMFs are transmitted bidirectionally by using the Generate Tone/Wait for Tone script functions on both scenario channels. Phone A on the first scenario channel uses a Skinny Make Call script function to initiate a call based on the Dial Plan settings. The Sk_Receive AnswerCall_HL procedure used by the second scenario channel is a wrapping of the common Skinny Wait Call and Skinny Answer Call script functions. Eventually, the call is terminated by phone A by using a Skinny End Call procedure.
Execution Settings	The corresponding scenario channel is configured to execute one loop during sustain time and to use one alias/channel.
Dial Plan	The registration names are defined by using a SEP7902A00[00001-] sequence generating expression. SK_SEP7902B0000001 is configured as call destination and no transfer destination needs to be configured.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The Sequential registration and the Fail if previously has failed options are selected.
Codec Settings	The G.711 codec is selected for the tone to be transmitted inband.
RTP Settings	The Enable media on this activity option is selected for media streaming to be performed between the emulated phones.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** SK_SEP7902B0000001 uses the same settings as SK_SEP7902A0000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. The registration names of the SK_SEP7902B0000001 - emulated phones are specified by using a SEP7902B00[00001-] sequence generating expression.

Advanced Call Features

This tests group is aimed at testing advanced call features provided by the Cisco CallManager, such as forward, conference calls, or hold/retrieve.

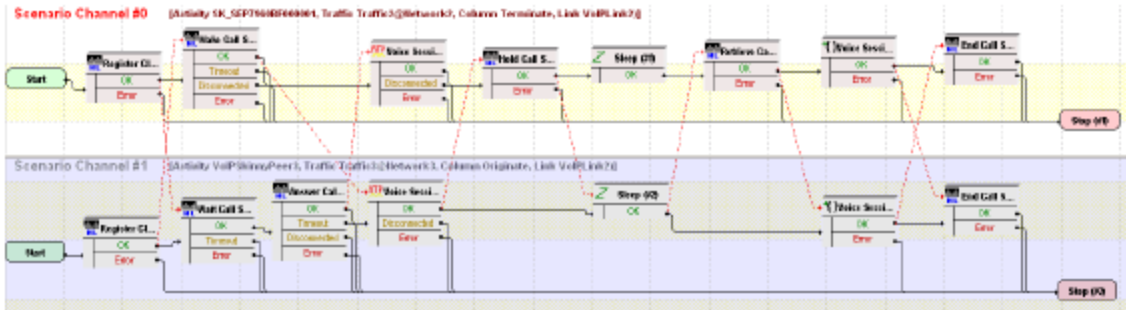
VoIPSkinnyPeer activities in this group are configured to emulate a number of five 7960-type Skinny phones, and tests execute a single loop during test sustain time.

 **Note:** The network settings for all tests described subsequently are the same, namely 1 IPv4 host with no emulated router.

SK_022_7960_SM_US_5_ChS_IPv4_Static_Hold_Resume

This test, illustrating a call hold and retrieve procedure, comprises two VoIPSkinny Peer activities: phone A emulated by SK_SEP7960AF000001 establishes a call with the phone B emulated SK_SEP7960BF000001, and then puts the call on hold and remains idle for the duration of the Sleep script function.

Eventually, phone A retrieves the call to B previously put on hold and performs another voice session.



SK_SEP7960AF000001 configured settings are described in the following table:

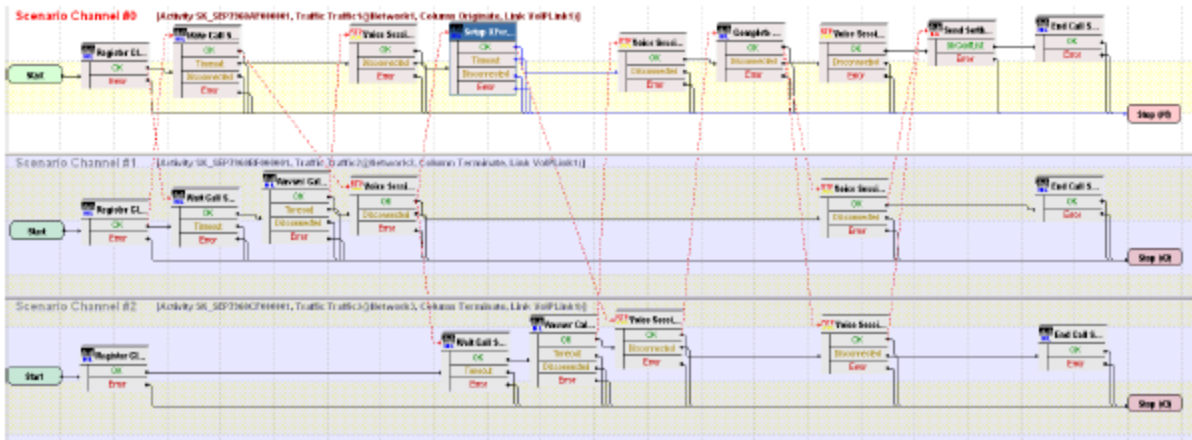
Category	Settings
Scenario Editor	The two channels test scenario is completely configured. Using Dial Plan settings, phone A on the first scenario channel establishes a call with phone B on the second channel. After putting the call on hold, phone A remains idle for the duration of the Sleep function, and then retrieves the call. Media streaming by using the Voice Session script function is performed each time two phones get connected.
Execution Settings	The corresponding scenario channel is configured to execute one loop during sustain time and to use one alias/channel.
Dial Plan	The registration names are defined by using a SEP7960AF0[00001-] sequence generating expression. SK_SEP7960BF000001 is configured as a call destination.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area.
Codec Settings	The codec settings can be left unchanged.
RTP Settings	Because the activity performs RTP streaming, the Enable media on this activity option is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: SK_SEP7960BF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. The registration names are defined by using a SEP7960BF0[00001-] sequence generating expression.

SK_031_7960_SM_US_5_ChS_IPv4_Static_List_Ad_Hoc_Conference

This test, illustrating a call conference procedure, comprises three VoIPSkinnyPeer activities: phone A emulated by SK_SEP7960AF000001 establishes a call with the phone B emulated SK_SEP7960BF000001 and performs media streaming. Using the SetUp Xfer and the Complete Xfer script functions, phone A establishes a conference call with phone C of SK_SEP7960CF000001, joining all phones in a conference.

After all the phones are connected in an Ad-Hoc conference, they perform media streaming. Eventually, phone A presses the SkConflist softkey showing all parties involved in the conference.



SK_SEP7960AF000001 configured settings are described in the following table:

Category	Settings
Scenario Editor	Using the Dial Plan settings, phone A on the first scenario channel establishes a call with phone B on the second channel. After performing a media session, phone A establishes a conference call with phone C and joins all calls by using a Complete Xfer script function. Eventually, phone A presses the SkConflist softkey showing all parties involved in the conference call.
Execution Settings	The corresponding scenario channel is configured to execute one loop during sustain time and to use one alias/channel.
Dial Plan	The registration names are defined by using a SEP7960AF0[00001-] sequence generating expression. SK_SEP7960BF000001 is configured as a call destination, while SK_SEP7960CF000001 is configured as call transfer destination.

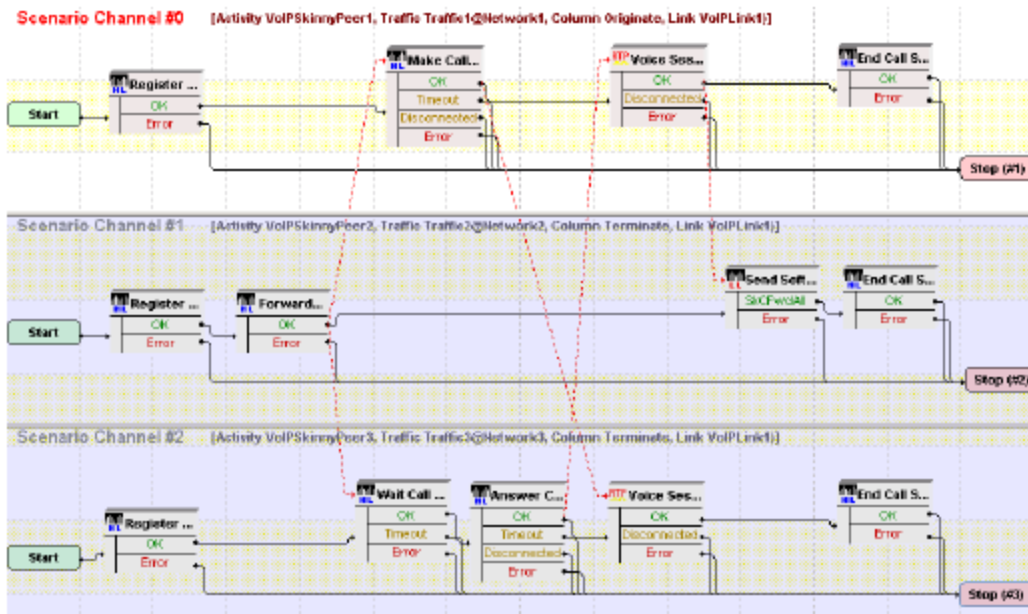
Skiny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area.
Codec Settings	The codec settings can be left unchanged.
RTP Settings	Because the activity performs RTP streaming, the Enable media on this activity option is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: SK_SEP7960BF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. Its registration names are defined by using a SEP7960BF0[00001-] sequence generating expression. SK_SEP7960CF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. Its registration names are defined by using a SEP7960CF0[00001-] sequence generating expression.

SK_032_7960_SM_US_5_Chs_IPv4_Static_Forward_All_Calls

This test, illustrating a call forwarding procedure, comprises three VoIPSkinnyPeer activities: phone A emulated by SK_SEP7960AF000001 originates a call to SK_SEP7960BF000001, while phone B emulated by SK_SEP7960BF000001 executes a Skinny ForwardAllCalls function, instructing the CallManager to forward the call to phone C emulated by SK_SEP7960CF000001.

After the call is established between phones A and C, media streaming is performed between the two parties.



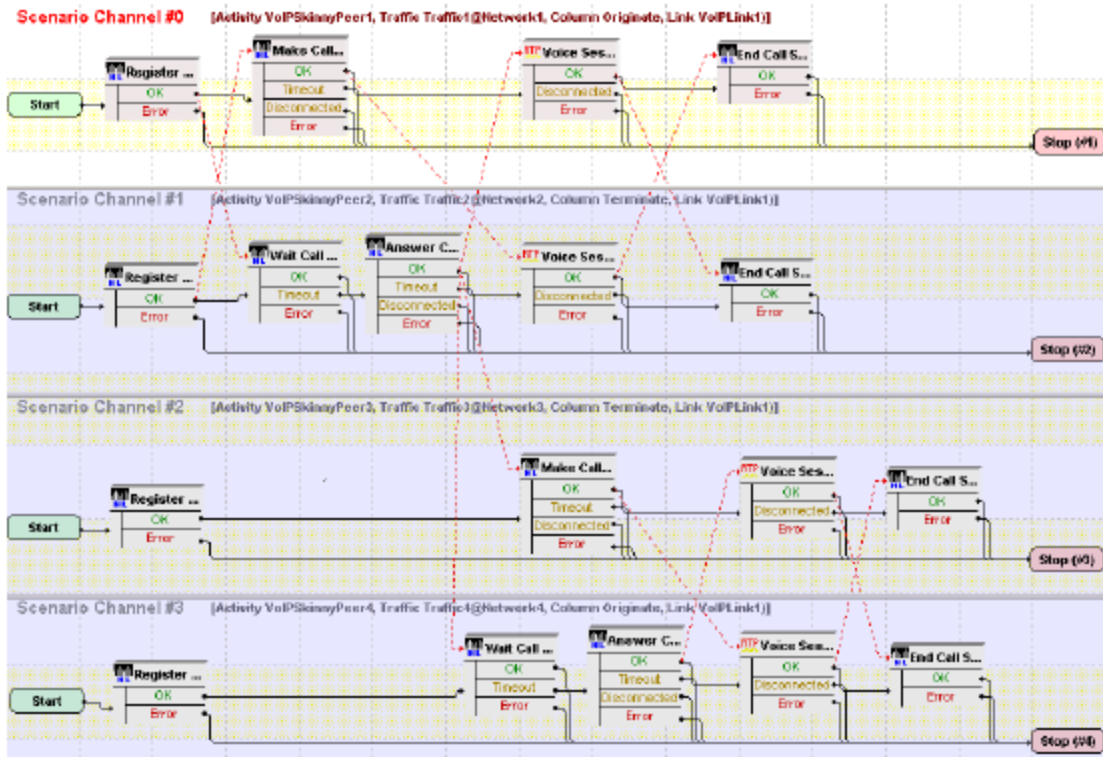
VoIPSkinnyPeer1 configured settings are described in the following table:

Category	Settings
Scenario Editor	The three channels test scenario is completely configured. Phone A on the first scenario channel phone attempts to establish a call with the phone B, which executes a Skinny ForwardAllCalls script function and thus instructs the CallManager to forward the call to phone C on the third channel. The Sk_Receive AnswerCall_HL procedure used by phone C to answer the incoming call is a wrapping of the common Skinny Wait Call and Skinny Answer Call script functions.
Execution Settings	The corresponding scenario channel is configured to execute once during test sustain time and to use one alias/voice channel.
Dial Plan	The registration names are defined by using a SEP7960AF0[00001-] sequence generating expression. The SK_SEP7960BF000001 activity is configured as call destination.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area.
Codec Settings	The default codec settings are used.
RTP Settings	Because the activity performs RTP streaming, the Enable media on this activity option is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: SK_SEP7960BF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not specify a call destination. Its registration names are defined by using a SEP7960BF0[00001-] sequence generating expression. SK_SEP7960CF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not need to specify a call destination activity, because it only terminates a call. Its registration names are defined by using a SEP7960CF0[00001-] sequence generating expression.

SK_033_7960_SM_US_5_ChS_IPv4_Static_Forward_Busy

This test, illustrating a call forwarding procedure, comprises four VoIPSkinnyPeer activities: after phone A emulated by SK_SEP7960AF400001 has established a call with phone B emulated by SK_SEP7960BF400001, phone C corresponding to SK_SEP7960CF400001 calls phone B. Because phone B is already involved in a call, configured CallManager settings determine the call to be forwarded to phone D corresponding to SK_SEP7960DF400001, which answers the call.



SK_SEP7960AF400001 configured settings are described in the following table:

Category	Settings
Scenario Editor	The four channels test scenario is completely configured.
Execution Settings	The corresponding scenario channel is configured to execute once during test sustain time and to use one alias/voice channel.
Dial Plan	The registration names are defined by using a SEP7960AF4[00001-] sequence generating expression. SK_SEP7960BF400001 is configured as a call destination.
Skiny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area.
Codec Settings	The codec settings can be left unchanged.
RTP Settings	Because the activity performs RTP streaming, the Enable media on this activity option is selected.

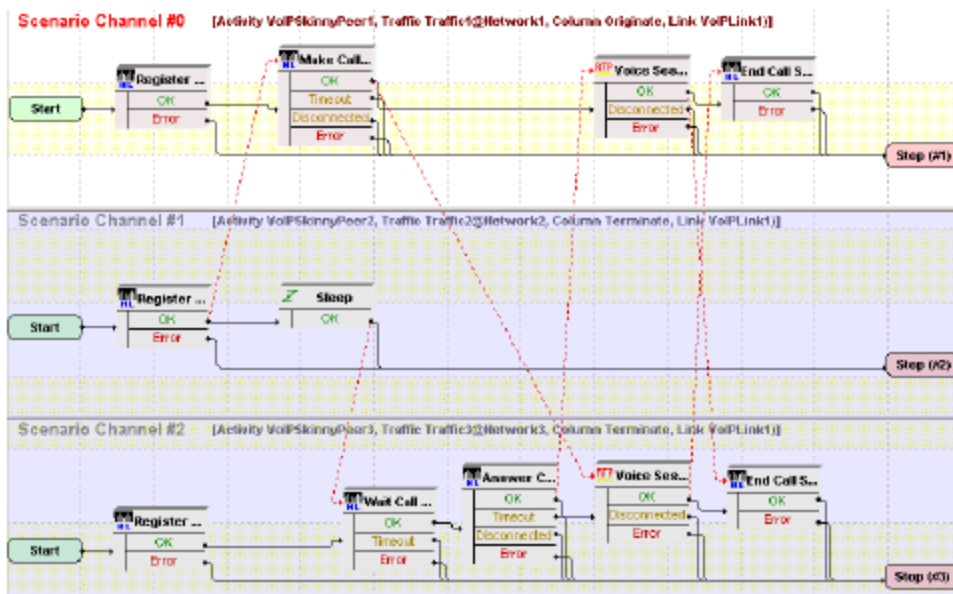
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.
----------------	---

Note: SK_SEP7960BF400001 uses the same settings as SK_SEP7960AF400001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. Its registration names are defined by using a SEP7960BF4[00001-] sequence generating expression. SK_SEP7960CF400001 uses the same settings as SK_SEP7960AF400001, except for the **Dial Plan** page, which specifies SK_SEP7960BF400001 as call destination, because it attempts to establish a call with it. Its registration names are defined by using a SEP7960CF4[00001-] sequence generating expression. SK_SEP7960DF400001, the final destination of the call initiated by SK_SEP7960CF400001, is configured the same as SK_SEP7960AF400001, except for the **Dial Plan** page, which does not need to specify any call or call transfer destination. Its registration names are defined by using a SEP7960DF4[00001-] sequence generating expression.

Important! For this test to execute correctly, call forwarding on a busy status for the Skinny phone B emulated by SK_SEP7960BF400001 needs to be configured on the Cisco CallManager.

SK_034_7960_SM_US_5_Chs_IPv4_Static_Forward_No_Answer


This test, illustrating a call forward procedure, comprises three VoIPSkinnyPeer activities: phone A emulated by SK_SEP7960AF500001 calls phone B emulated by SK_SEP7960BF500001, which forwards the call to phone C corresponding to SK_SEP7960CF500001.




SK_SEP7960AF500001 configured settings are described in the following table:

Category	Settings
----------	----------

Scenario Editor	The three channels test scenario is completely configured. Using Dial Plan settings, phone A of SK_SEP7960AF500001 calls phone B of SK_SEP7960BF500001, which does not answer the call and forwards the call to phone C of SK_SEP7960CF500001.
Execution Settings	The corresponding scenario channel is configured to execute once during test sustain time and to use one alias/voice channel.
Dial Plan	The registration names are defined by using a SEP7960AF4[00001-] sequence generating expression. SK_SEP7960BF400001 is configured as a call destination.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area.
Codec Settings	The codec settings can be left unchanged.
RTP Settings	Because the activity performs RTP streaming, the Enable media on this activity option is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

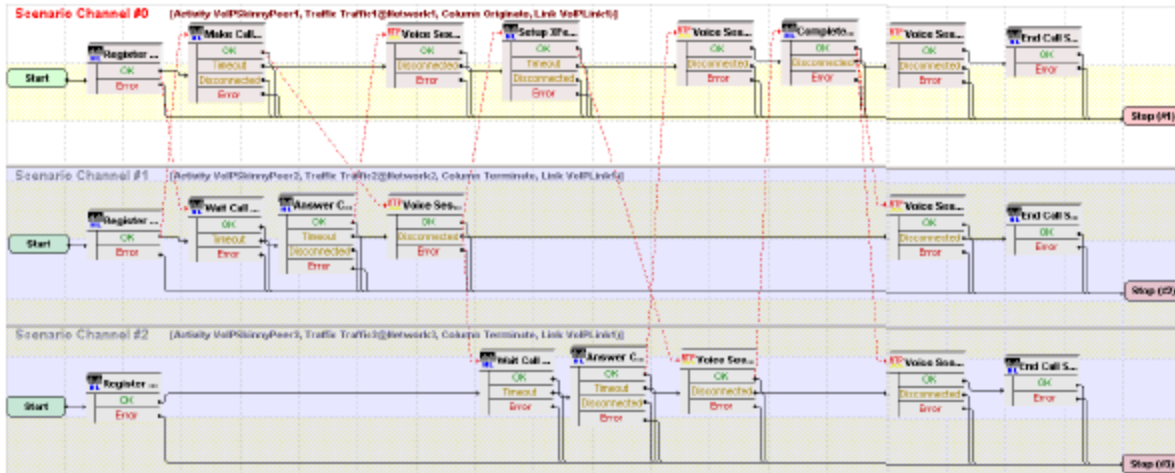
 **Note:** SK_SEP7960BF500001 uses the same settings as SK_SEP7960AF500001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. Its registration names are defined by using a SEP7960BF5[00001-] sequence generating expression. SK_SEP7960CF500001 uses the same settings as SK_SEP7960AF500001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. Its registration names are defined by using a SEP7960CF5[00001-] sequence generating expression.

 **Important!** For this test to execute correctly, call forwarding on a no answer status for the Skinny phone emulated by SK_SEP7960BF500001 needs to be configured on the Cisco CallManager.

SK_026_7960_SM_US_5_ChS_IPv4_Static_Ad_hoc_Conference

This test, illustrating an Adhoc conference procedure, comprises three VoIPSkinnyPeer activities: phone A emulated by SK_SEP7960AF000001 originates a call to phone B emulated by SK_SEP7960BF000001, and then creates a conference call to phone C of SK_SEP7960CF000001 and finally joins all parties in an Adhoc conference.

After the conference is established, all parties perform voice sessions between them.



SK_SEP7960AF000001 configured settings are described in the following table:

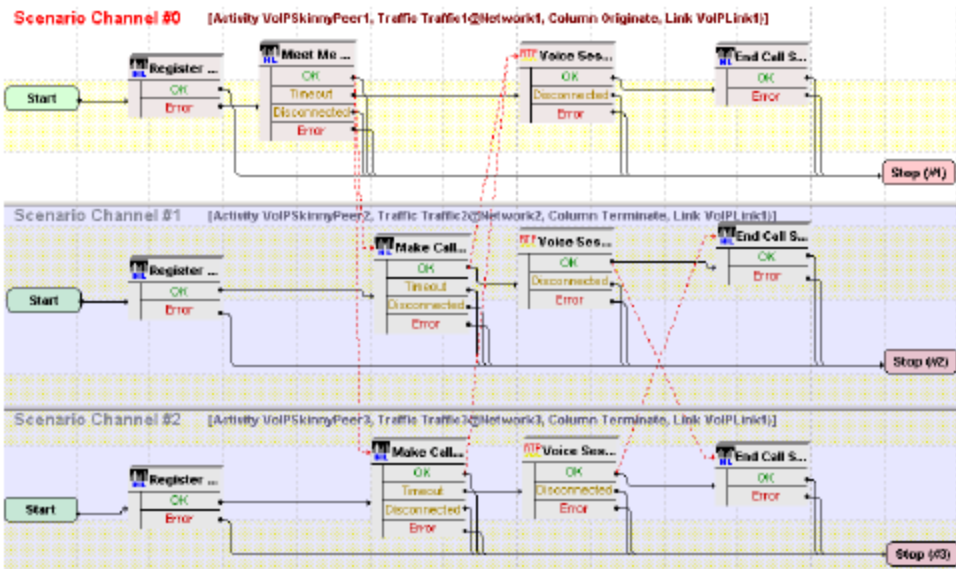
Category	Settings
Scenario Editor	The test scenario is completely configured.
Execution Settings	The corresponding scenario channel is configured to execute once during test sustain time and to use one alias/voice channel.
Dial Plan	The registration names are defined by using a SEP7960AF0[00001-] sequence generating expression. The SK_SEP7960BF000001 activity is configured as call destination, SK_SEP7960CF000001 is configured as call transfer/conference destination.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area.
Codec Settings	The codec settings can be left unchanged.
RTP Settings	Because the activity performs RTP streaming, the Enable media on this activity option is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: SK_SEP7960BF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. Its registration names are defined by using a SEP7960BF0[00001-] sequence generating expression. SK_SEP7960CF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does specify any call or call transfer destination. Its registration names are defined by using a SEP7960CF0[00001-] sequence generating expression.

SK_027_7960_SM_US_5_ChS_IPv4_Static_MeetMe_Conference

This test, illustrating a MeetMe conference procedure, comprises three VoIPSkinnyPeer activities: phone A emulated by SK_SEP7960AF000001 establishes a call to a specified conference number, while phones B and C, corresponding to SK_SEP7960BF000001 and SK_SEP7960CF000001 respectively, join the conference by also establishing calls to that number.


After the conference connection is established, all parties perform voice sessions.



SK_SEP7960AF000001 configured settings are described in the following table:

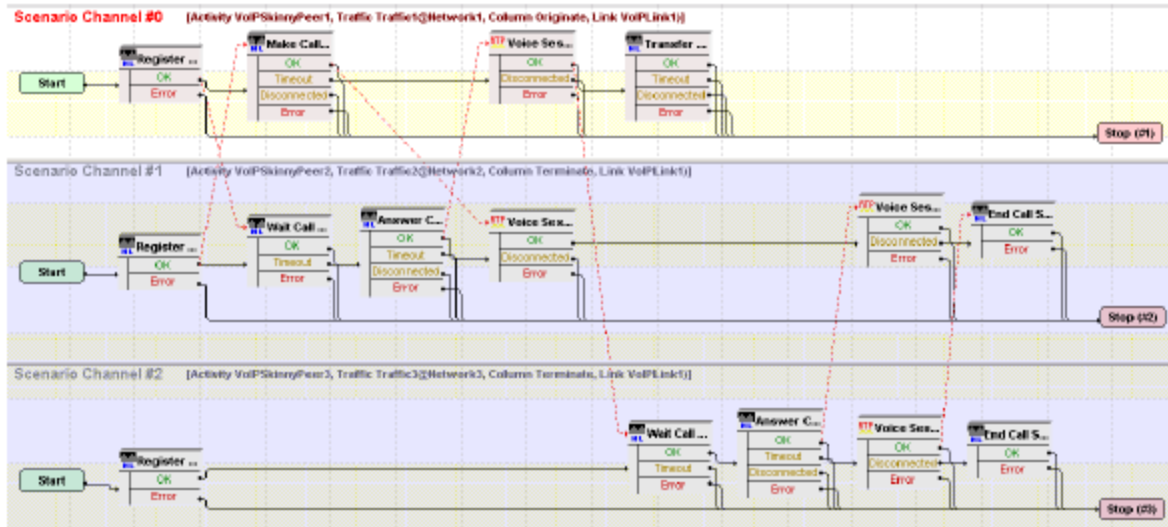
Category	Settings
Scenario Editor	The three channel test scenario is completely configured. The Skinny Meet-Me script function used by phone A on the first scenario channel sets up the conference by dialing to the pre-configured number ('6000' in the case of this sample test), while the Skinny Make Call functions on the other channels dial to the same number to join the conference.
Execution Settings	The corresponding scenario channel is configured to execute once during test sustain time and to use one alias/voice channel.

Scenario Editor	The three channel test scenario is completely configured. After registration of the phones A, B, and C with the CallManager specified by the Dial Plan settings, phone A connects to phone B, while phone C connects to phone B. Phone B executes the Skinny Send Softkey function with the SkJoin value that joins the calls.
Execution Settings	The corresponding scenario channel is configured to execute once during test sustain time and to use one alias/voice channel.
Dial Plan	The registration names are defined by using a SEP7960AF0[00001-] sequence generating expression. SK_SEP7960BF000001 is configured as call destination.
Skippy Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area.
Codec Settings	The default codec settings are used.
RTP Settings	Because the activity performs RTP streaming, the Enable media on this activity option is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** SK_SEP7960BF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does specify any call destination or call transfer destination. Its registration names are defined by using a SEP7960BF0[00001-] sequence generating expression. SK_SEP7960CF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which specifies SK_SEP7960BF000001 as call destination. Its registration names are defined by using a SEP7960CF0[00001-] sequence generating expression.

SK_024_7960_SM_US_5_Ch5_IPv4_Static_Blind_Transfer

This test, illustrating a blind transfer procedure, comprises three VoIPSkinnyPeer activities: phone A emulated by SK_SEP7960AF000001 establishes a call to phone B emulated by SK_SEP7960BF000001, and then transfers the call directly to phone C of SK_SEP7960CF000001. After the call transfer occurs, phone A goes on hook, and the two parties involved in the call, phone B and phone C, perform voice sessions.



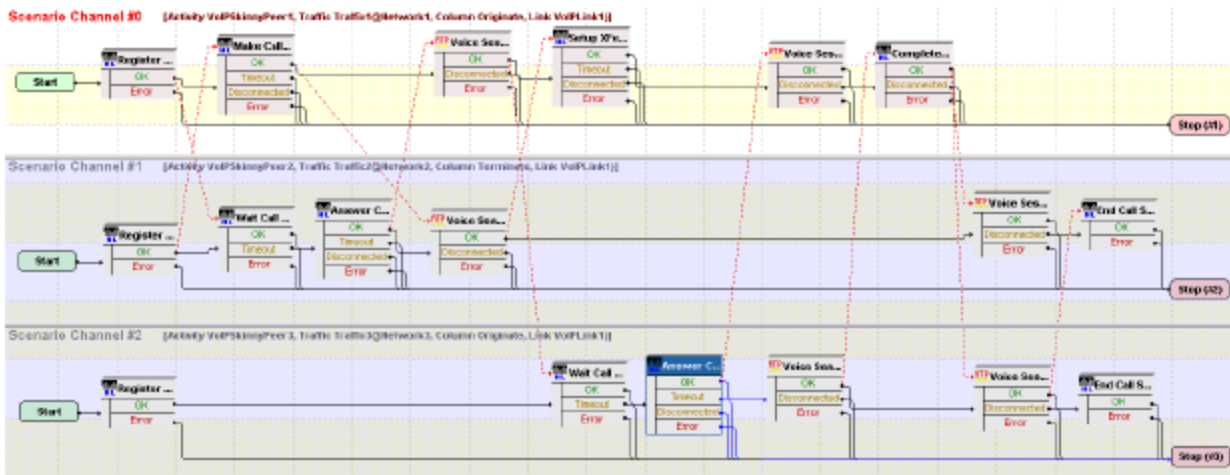
SK_SEP7960AF000001 configured settings are described in the following table:

Category	Settings
Scenario Editor	The three channel test scenario channel is completely configured. Phone A on the first scenario channel phone calls phone B by using a Skinny Make Call script function, while B answers the call by using the common Skinny Wait Call and Skinny Answer Call script functions. After the execution of a voice session between phones A and B, phone A transfers the call to phone C by executing a Skinny Transfer script function with the Blind Transfer option selected. After the phones are connected, phones B and C perform media streaming.
Execution Settings	The corresponding scenario channel is configured to use one alias/channel.
Dial Plan	The registration names are defined by using a SEP7960AF0[00001-] sequence generating expression. SK_SEP7960BF000001 is configured as a call destination, while SK_SEP7960CF000001 is configured as a call transfer/conference destination.
Skiny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area.
Codec Settings	The default codec settings are used.
RTP Settings	Because the activity performs RTP streaming, the Enable media on this activity option is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: SK_SEP7960BF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. Its registration names are defined by using a SEP7960BF0[00001-] sequence generating expression. SK_SEP7960CF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not specify any call destination or call transfer destination. Its registration names are defined by using a SEP7960CF0[00001-] sequence generating expression.

SK_023_7960_SM_US_5_ChS_IPv4_Static_Transfer_with_Consultation


This test, illustrating a transfer procedure with consultation, comprises three VoIPSkinnyPeer activities: phone A emulated by SK_SEP7960AF000001 establishes a call to phone B emulated by SK_SEP7960BF000001, and then transfers the call to phone C of SK_SEP7960CF000001. The only difference to the previously described test is that phone A performs a voice session (consultation) with phone C before transferring the call. After the call transfer occurs, phone A goes on hook, and the two parties involved in the call, phones B and C, perform voice sessions.



SK_SEP7960AF000001 configured settings are described in the following table:

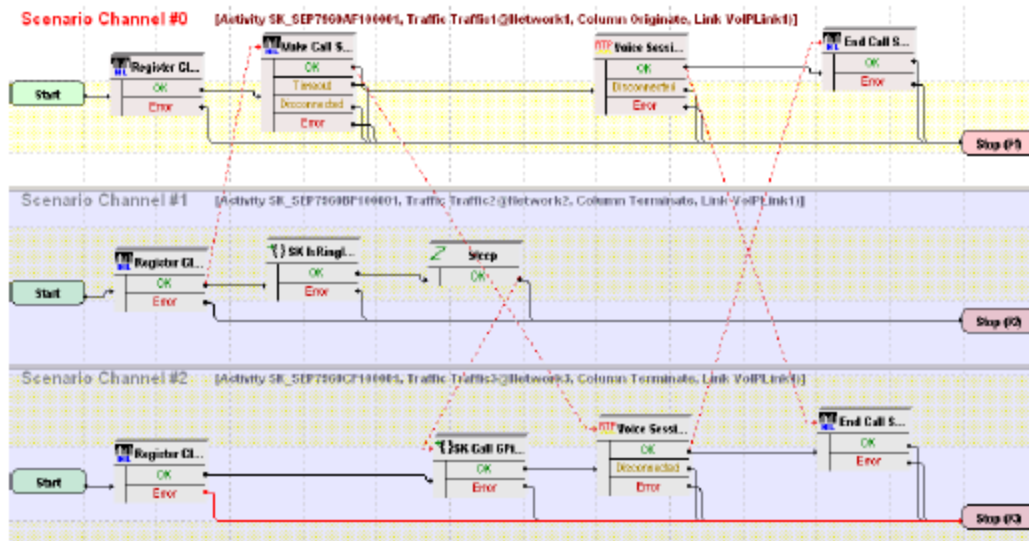
Category	Settings
Scenario Editor	The three channel test scenario is completely configured. Phone A on the first scenario channel phone calls phone B by using a Skinny Make Call script function, while B answers the call by using the common Skinny Wait Call and Skinny Answer Call script functions. After the execution of a voice session between phones A and B, phone A initiates the transfer the call to phone C by executing a Skinny Setup Xfer followed by a Skinny Complete Xfer script function. After the phones are connected, phones B and C perform media streaming.
Execution Settings	The corresponding scenario channel is configured to use one alias/channel.

Scenario Editor	The three channel test scenario is completely configured. Phone A on the first scenario channel calls phone B by using a Skinny Make Call script function, and phone B answers the call by using the common Skinny Wait Call and Skinny Answer Call script functions. After the phones are connected, the phones perform media streaming. Phone C also calls phone B which answer the call, media streaming is performed between the connected phones. Phone B then transfers the call to phone A by using a SendSoftkey (SkDirTrfr) script function. After the phones are connected, phones A and C perform media streaming.
Execution Settings	The corresponding scenario channel is configured to use one alias/channel.
Dial Plan	The registration names are defined by using a SEP7960AF0[00001-] sequence generating expression. SK_SEP7960BF000001 is configured as a call destination.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area.
Codec Settings	The default codec settings are used.
RTP Settings	Because the activity performs RTP streaming, the Enable media on this activity option is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

 **Note:** SK_SEP7960BF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. Its registration names are defined by using a SEP7960BF0[00001-] sequence generating expression. SK_SEP7960CF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not specify any call destination or call transfer destination. Its registration names are defined by using a SEP7960CF0[00001-] sequence generating expression.

SK_036_7960_SM_US_5_ChS_IPv4_Static_Call_Group_Pickup

This test, illustrating a call pickup procedure, comprises three VoIPSkinnyPeer activities: phone A emulated by SK_SEP7960AF100001 establishes a call to phone B emulated by SK_SEP7960BF100001, which does not answer the call. Phone C of SK_SEP7960CF100001 dials the number of the call group comprising phones A and B and picks up the call. After the phones are connected, the two parties involved in the call, phones A and C, perform a voice session.



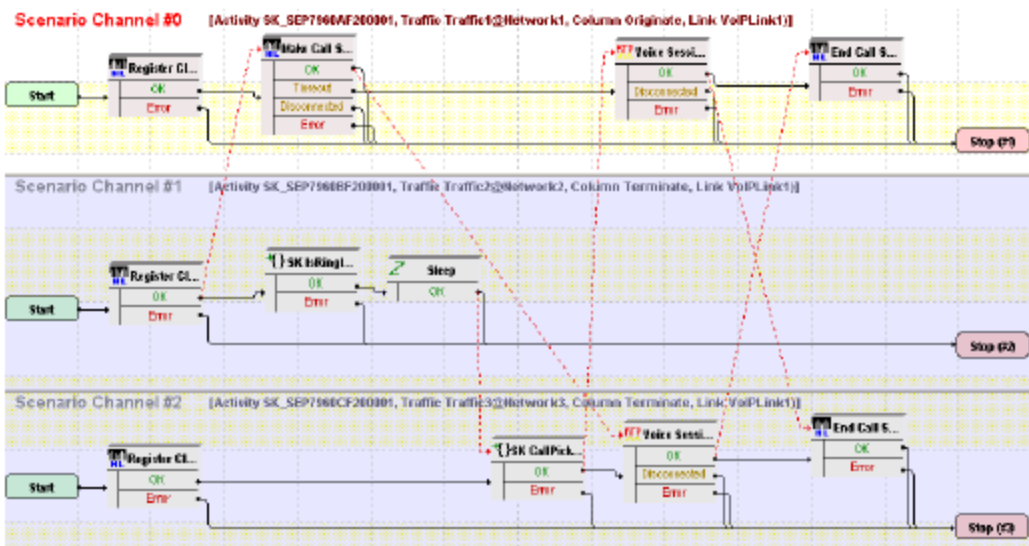
SK_SEP7960AF100001 configured settings are described in the following table:

Category	Settings
Scenario Editor	The three channel test scenario is completely configured. On the Cisco CallManager, phones A and B are configured as part of one call group, while phone C is configured as part of another. Phone A on the first scenario channel calls phone B by using a Skinny Make Call script function, with phone B not answering the call. Using a Skinny Call GPickUp procedure, phone C dials the group number of phones A and B and is able to pick up the call addressed to phone B. After the phones are connected, phones A and C perform media streaming.
Execution Settings	The corresponding scenario channel is configured to use one alias/channel.
Dial Plan	The registration names are defined by using a SEP7960AF0[00001-] sequence generating expression. SK_SEP7960BF000001 is configured as a call destination.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area.
Codec Settings	The default codec settings are used.
RTP Settings	Because the activity performs RTP streaming, the Enable media on this activity option is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: SK_SEP7960BF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. Its registration names are defined by using a SEP7960BF0[00001-] sequence generating expression. SK_SEP7960CF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not specify any call destination or call transfer destination. Its registration names are defined by using a SEP7960CF0[00001-] sequence generating expression.

SK_037_7960_SM_US_5_Chs_IPv4_Static_Call_Pickup

This test, illustrating a call pickup procedure, comprises three VoIPSkinnyPeer activities: phone A emulated by SK_SEP7960AF200001 establishes a call to phone B emulated by SK_SEP7960BF200001, which does not answer the call. Phone C of SK_SEP7960CF200001 presses the Call Pickup softkey and picks up the call. After the phones are connected, the two parties involved in the call, phones A and C, perform a voice session.



SK_SEP7960AF200001 configured settings are described in the following table:

Category	Settings
Scenario Editor	The three channel test scenario is completely configured. On the Cisco CallManager, phones A, B, and C are configured as part of the same call group. Phone A on the first scenario channel calls phone B by using a Skinny Make Call script function, with phone B not answering the call. Phone C presses the Call Pickup softkey and picks up the call addressed to phone B. After the phones are connected, phones A and C perform media streaming.
Execution Settings	The corresponding scenario channel is configured to use one alias/channel.
Dial Plan	The registration names are defined by using a SEP7960AF0[00001-] sequence generating expression. SK_SEP7960BF000001 is configured as a call destination.

Skinnny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area.
Codec Settings	The default codec settings are used.
RTP Settings	Because the activity performs RTP streaming, the Enable media on this activity option is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

Note: SK_SEP7960BF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not specify any call or call transfer destination. Its registration names are defined by using a SEP7960BF0[00001-] sequence generating expression. SK_SEP7960CF000001 uses the same settings as SK_SEP7960AF000001, except for the **Dial Plan** page, which does not specify any call destination or call transfer destination. Its registration names are defined by using a SEP7960CF0[00001-] sequence generating expression.

Mixed Skinny and SIP - SIP UAs

This group of test samples emulates Skinny and SIP phones that register with a Cisco CallManager as Skinny and SIP phones respectively and establish calls with each other. These samples are intended for running against a Cisco CallManager 5.x supporting the registration of Cisco SIP phones.

Test samples comprise both signaling-only and signaling with media calls.

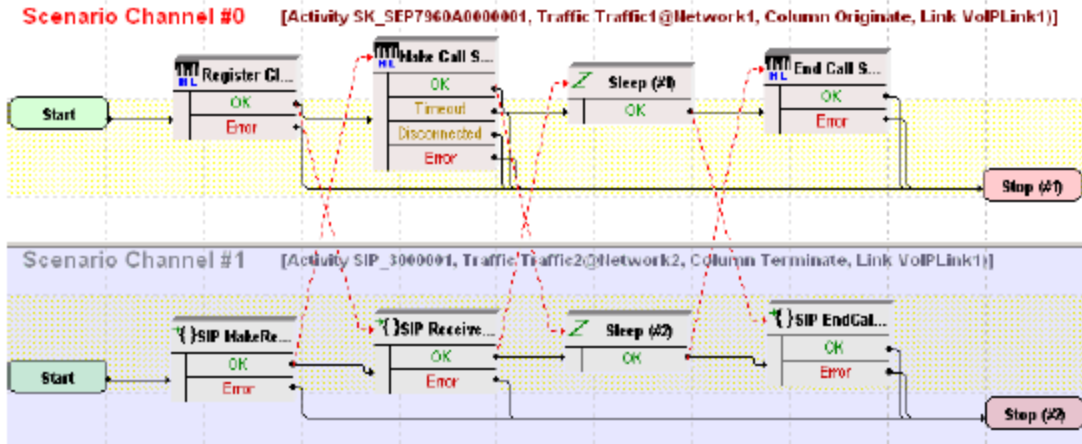
Note: For this tests category, all emulated SIP UAs are configured by using consecutive IP addresses, the same port and consecutive phone number values.

MIX_023_7960_S0_US_3000_ChS_IPv4_Static_SK_to_SIP_Call_10s

This test illustrates a mixed Skinny to SIP call procedure without media streaming. A number of 7960 Skinny phones emulated by the SK_SEP7960A0000001 activity register with the Cisco CallManager, and then establish calls with the SIP phones emulated by the SIP_3000001 activity and registered with the Cisco CallManager as SIP phones. After the call is established, the connection is kept up for the duration of the Sleep functions, configured to 10 seconds.

The call procedure is executed once for the sustain time duration of the test.

The underlying two-channel test scenario involving Skinny phone A and SIP phone B is shown in the following image:



To be able to run the test on your computer, select the Skinny configuration settings (SK_SEP7960A000001) and the SIP settings (SIP_3000001) described in the following two tables respectively:

Category	Settings
Scenario Editor	The test scenario is completely configured and comprises the phones registration followed by a call establishment procedure, a call sustain time without media streaming, and the call tear down. On the call originating Skinny scenario channel, phone A registers with the Cisco CallManager as a Skinny phone and originates a call by using the Skinny Make Call script function. The call duration is configured to 10 seconds by using the Sleep function. The last scenario channel procedure is the Skinny End Call function that terminates the call. On the call terminating SIP scenario channel, phone B registers with the Cisco CallManager as an SIP phone, and then executes an SIP Receive Call procedure to answer the call. The Sleep function ensures a call duration of 10 seconds. The call is terminated by using the SIP EndCall Receive procedure for the call terminating side.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time and to use one alias/channel.
Dial Plan	The Skinny registration names are defined by using a SEP7960A00[00001-] sequence generating expression. SIP_3000001:[5060-] is configured as call destination.
Skiny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skinny traffic is selected.
Codec Settings	The default codec settings are used.

RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skinny channels.

Category	Settings
Scenario Editor	The test scenario is completely configured and comprises the phones registration followed by a call establishment procedure, a call sustain time without media streaming, and the call tear down. On the call originating Skinny scenario channel, phone A registers with the Cisco CallManager as a Skinny phone and originates a call by using the Skinny Make Call script function. The call duration is configured to 10 seconds by using the Sleep function. The last scenario channel procedure is the Skinny End Call function that terminates the call. On the call terminating SIP scenario channel, phone B registers with the Cisco CallManager as an SIP phone, and then executes an SIP Receive Call procedure to answer the call. The Sleep function ensures a call duration of 10 seconds. The call is terminated by using the SIP EndCall Receive procedure for the call terminating side.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time. The emulated SIP phones use the consecutive IP addresses, the same port, and consecutive phone numbers.
Dial Plan	SIP endpoints phone numbers are defined by using a 30[00001-] sequence generating expression. Because this channel only terminates a call, no call destination or call transfer destination needs to be configured.
SIP Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Port field in the SIP Settings area is specified by using a [5060-] sequence generating expression. In the Use Server area that needs to specify an SIP proxy server address and port, the address is that of the Cisco CallManager, and the port is 5060. Outbound proxy and registrar functionalities are also configured. The industry-standard Class 3 (0x60) TOS/DSCP setting for SIP traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.

Other Settings	<p>The IP version preference is set to IPv4, and the following global scenario variables are initialized and can be used by all SIP channels:</p> <ul style="list-style-type: none"> • VoIP_Var0 is set to a 7960BBBB[0000-] sequence generating expression. This variable is used by the SIP Make Registration procedure in the SIP Send Request (INVITE) script function to register the SIP phone with a Cisco CallManager. • VOIP_Var4 is set to '1.' This variable is used by the SIP Make Registration procedure to test whether the registration is executed for every loop (for a value of '1,') or only for the first loop (for a value of '2.')
----------------	---

MIX_024_7960_SO_US_3000_ChS_IPv4_Static_SK_to_SIP_Call_3min

This test is similar to the previous one, with the only difference that the call duration is three minutes, configured by using the Sleep script function.

MIX_025_7960_SO_US_3000_ChS_IPv4_Static_SK_to_SIP_Call_30_min

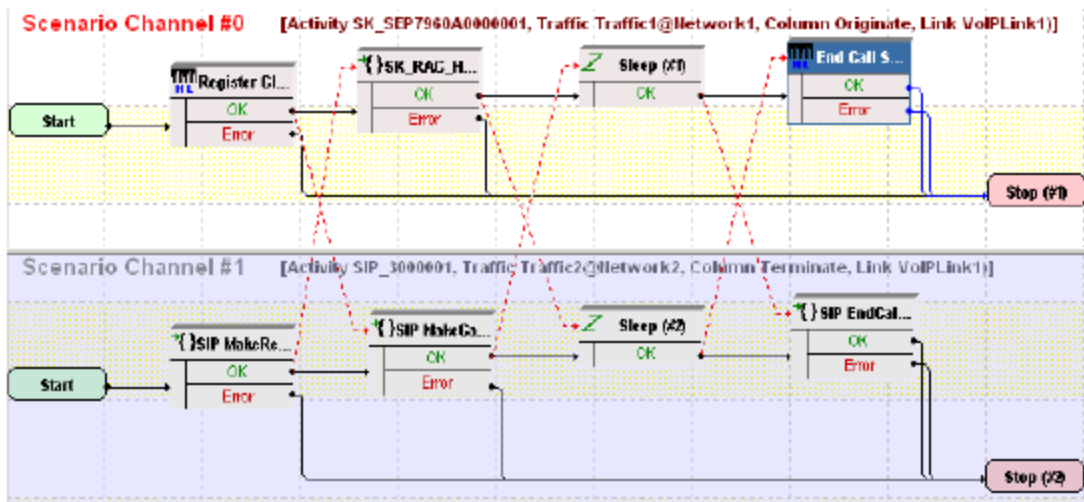
This test is similar to the previous one, with the only difference that the call duration is 30 minutes, configured by using the Sleep script function.

MIX_026_7960_SO_US_3000_ChS_IPv4_Static_SIP_to_SK_Call_10s

This test illustrates a mixed SIP to Skinny call procedure without media streaming. A number of 7960 SIP phones emulated by the SIP_3000001 activity register with the Cisco CallManager, and then establish calls with the Skinny phones emulated by the SK_SEP7960A0000001 activity and registered with the Cisco CallManager as Skinny phones. After establishing the call, a call duration of 10 seconds is configured by using Sleep script functions on both scenario channels.

The call procedure is executed once for the sustain time duration of the test.

The underlying two-channel test scenario involving Skinny phone A and SIP phone B is shown in the following image:



To be able to run the test on your computer, select the Skinny configuration settings (SK_SEP7960A0000001) and the SIP settings (SIP_3000001) described in the following two tables respectively:

Category	Settings
Scenario Editor	The test scenario is completely configured and comprises the phones registration followed by a call establishment procedure, a call sustain time without media streaming, and the call tear down. On the call originating scenario channel, SIP phone B registers with the Cisco CallManager as an SIP device and originates a calls towards the Skinny phone A. The established call is maintained active for a duration of 10 seconds by using the Sleep script function. The last procedure is a call termination procedure for the receiving side. On the call terminating scenario channel, Skinny phone A registers with the Cisco CallManager and executes a Sk_Receive AnswerCall_HL procedure – a wrapping of the common Skinny Wait Call and Skinny Answer Call script functions – to answer the call. The established call is maintained active for a duration of 10 seconds by using the Sleep script function. The call is terminated by phone A by using the Skinny End Call script function.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time and to use one alias/channel.
Dial Plan	The Skinny registration names are defined by using a SEP7960A00[00001-] sequence generating expression. Because this activity is terminating a call, no call destination needs to be configured.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skinny traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skinny channels.

Category	Settings
----------	----------

Scenario Editor	The test scenario comprises the phones registration followed by a call establishment procedure, a call sustain time without media streaming, and the call tear down. On the call originating scenario channel, SIP phone B registers with the Cisco CallManager as an SIP device and originates a calls towards the Skinny phone A. The established call is maintained active for a duration of 10 seconds by using the Sleep script function. The last procedure is a call termination procedure for the receiving side. On the call terminating scenario channel, Skinny phone A registers with the Cisco CallManager and executes a Sk_Receive AnswerCall_HL procedure – a wrapping of the common Skinny Wait Call and Skinny Answer Call script functions – to answer the call. The established call is maintained active for a duration of 10 seconds by using the Sleep script function. The call is terminated by phone A by using the Skinny End Call script function.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time. The emulated SIP phones use the consecutive IP addresses, the same port, and consecutive phone numbers.
Dial Plan	SIP endpoints phone numbers are defined by using a 30[00001-] sequence generating expression. SK_SEP7960A0000001 is configured as call destination.
SIP Settings	The Enable signaling on this activity option is selected for the SIP functions to be executed. The Port field in the SIP Settings area is specified by using a [5060-] sequence generating expression. In the Use Server area that needs to specify an SIP proxy server address and port, the address is that of the Cisco CallManager, and the port is 5060. Outbound proxy and registrar functionalities are also configured. The industry-standard Class 3 (0x60) TOS/DSCP setting for SIP traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because this activity performs no media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and the following global scenario variables are initialized and can be used by all SIP channels: <ul style="list-style-type: none"> VoIP_Var0 is set to a 7960BBBB[0000-] sequence generating expression. This variable is used by the SIP Make Registration procedure in the SIP Send Request (INVITE) script function to register the SIP phone with a Cisco CallManager. VOIP_Var4 is set to '1.' This variable is used by the SIP Make Registration procedure to test whether the registration is executed for every loop (for a value of '1,') or only for the first loop (for a value of '2.')

MIX_027_7960_SO_US_3000_ChS_IPv4_Static_SIP_to_SK_Call_3min

This test is similar to the previous one, with the only difference that the call duration is three minutes, configured by using the Sleep script function.

MIX_028_7960_SO_US_3000_ChS_IPv4_Static_SIP_to_SK_Call_30min

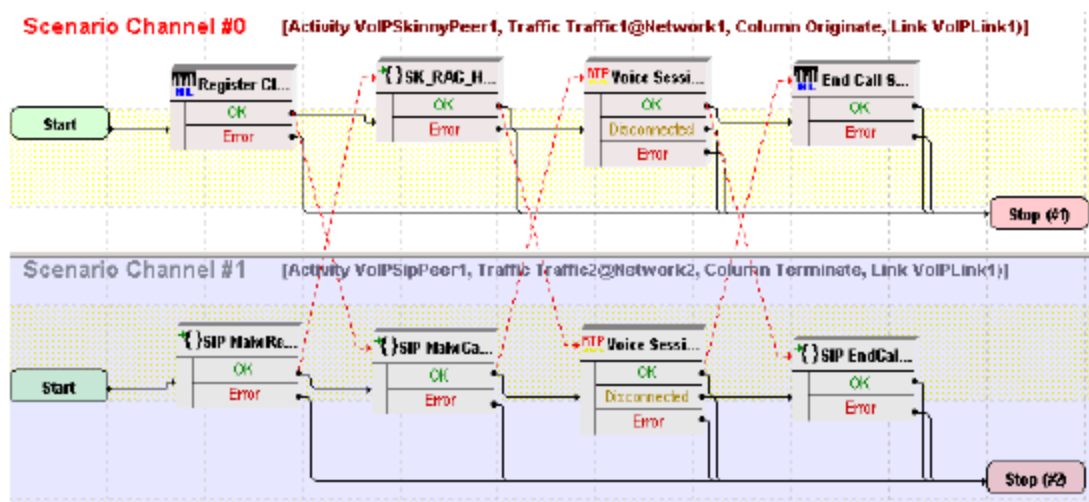
This test is similar to the previous one, with the only difference that the call duration is 30 minutes, configured by using the Sleep script function.

MIX_020_7960_SM_US_900_ChS_IPv4_Static_SIP_to_SK_Call_Voice_10s

This test illustrates a mixed SIP to Skinny call procedure with media streaming. A number of 7960 SIP phones emulated by the SIP_3000001 activity register with the Cisco CallManager, and then establish calls with the Skinny phones emulated by the SK_SEP7960A0000001 activity and registered with the CallManager. After establishing the call, the phones use the Voice Session script function for bidirectional media streaming.

The call procedure is repeated once for the sustain time duration of the test.

The underlying two-channel test scenario involving Skinny phone A and SIP phone B is shown in the following image:



To be able to run the test on your computer, select the Skinny configuration settings (SK_SEP7960A0000001) and the SIP settings (SIP_3000001) described in the following two tables respectively:

Category	Settings
Scenario Editor	The test scenario is completely configured and comprises the phones registration followed by a call establishment procedure, bidirectional media streaming and call tear down. On the call originating scenario channel, SIP phone B registers with the CallManager as an SIP phone and originates a call. Media is exchanged by using the Voice Session script function. The last procedure is a call termination procedure for the receiving side. On the call terminating Skinny scenario channel, Skinny phone A registers with the CallManager and executes a Sk_Receive AnswerCall_HL procedure – a wrapping of the common Wait Call and Answer Call script functions – to answer the call. The Voice Session function is used to perform bidirectional media streaming. The call termination is started by phone A by using the Skinny End Call script function.

Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time and to use one alias/channel.
Dial Plan	The Skinny registration names are defined by using a SEP7960A00[00001-] sequence generating expression. Because this activity is terminating a call, no call destination and no transfer destination needs to be configured.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skinny traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is selected. The industry-standard Express Forwarding (0xA0) TOS/DSCP setting for RTP traffic is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skinny channels.

Category	Settings
Scenario Editor	The test scenario is completely configured and comprises the phones registration followed by a call establishment procedure, bidirectional media streaming and call tear down. On the call originating scenario channel, SIP phone B registers with the CallManager as an SIP phone and originates a call. Media is exchanged by using the Voice Session script function. The last procedure is a call termination procedure for the receiving side. On the call terminating Skinny scenario channel, Skinny phone A registers with the CallManager and executes a Sk_Receive AnswerCall_HL procedure – a wrapping of the common Wait Call and Answer Call script functions – to answer the call. The Voice Session function is used to perform bidirectional media streaming. The call termination is started by phone A by using the Skinny End Call script function.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time and to use one alias/channel. The emulated SIP phones use consecutive IP addresses, the same port, and consecutive phone numbers.
Dial Plan	SIP endpoints phone numbers are defined by using a 30[00001-] sequence generating expression. SK_SEP7960A0000001 is configured as call destination.
SIP Settings	The Enable signaling on this activity option is selected for the SIP functions to be executed. The Port field in the SIP Settings area is specified by using a [5060-] sequence generating expression. In the Use Server area that needs to specify an SIP proxy server address and port, the address is that of the Cisco CallManager, and the port is 5060. Outbound proxy and registrar functionalities are selected.

Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is selected. In the Execution Settings page, the SIP phones are configured to use consecutive IPs and the same port. Hence, the RTP Port field is specified by using a single 10000 value.
Other Settings	The IP version preference is set to IPv4, and the following global scenario variables are initialized and can be used by all SIP channels: <ul style="list-style-type: none"> • VoIP_Var0 is set to a 7960BBBB[0000-] sequence generating expression. This variable is used by the SIP Make Registration procedure in the SIP Send Request (INVITE) script function to register the SIP phone with a Cisco CallManager. • VOIP_Var4 is set to '1.' This variable is used by the SIP Make Registration procedure to test whether the registration is executed for every loop (for a value of '1,') or only for the first loop (for a value of '2.')

MIX_021_7960_SM_US_900_ChS_IPv4_Static_SIP_to_SK_Call_Voice_3min

This test is similar to the previous one, with the only difference that the voice session has a duration of three minutes. The longer call duration is obtained by playing the selected wave file continuously for three minutes (**Advanced Playback Settings** tab of the Voice Session script function).

MIX_022_7960_SM_US_900_ChS_IPv4_Static_SIP_to_SK_Bulk_Call_Voice_30_min

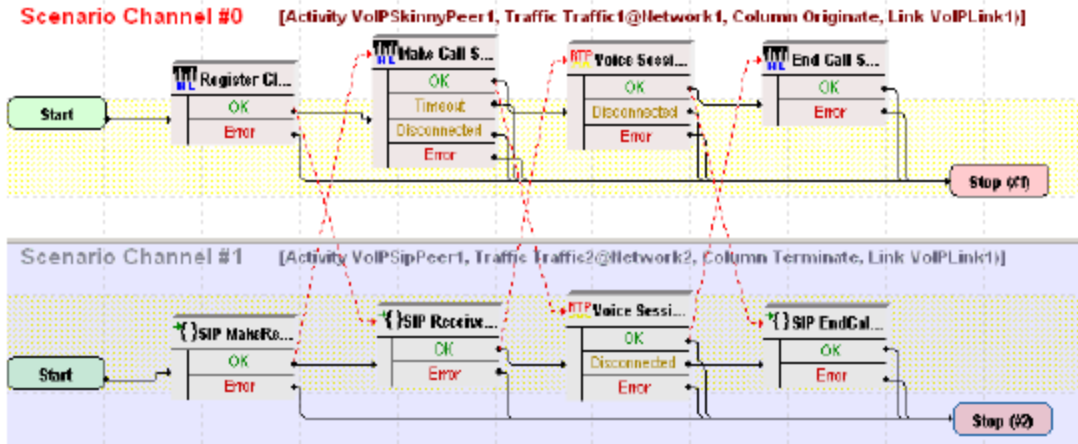
This test is similar to the previous one, with the only difference that the voice session has a duration of 30 minutes. The longer call duration is obtained by playing the selected wave file continuously for 30 minutes (**Advanced Playback Settings** tab of the Voice Session script function).

MIX_017_7960_SM_US_900_ChS_IPv4_Static_SK_to_SIP_Call_Voice_10s

This test illustrates a mixed Skinny to SIP call procedure with media streaming. A number of 7960 Skinny phones emulated by the SK_SEP7960A0000001 activity register with the Cisco CallManager, and then establish calls with the SIP phones emulated by the SIP_3000001 activity and registered with the CallManager as SIP phones. After establishing the call, the phones use the Voice Session script function for bidirectional media streaming.

The call procedure is repeated once for the sustain time duration of the test.

The underlying two-channel test scenario involving Skinny phone A and SIP phone B is shown in the following image:



To be able to run the test on your computer, select the Skinny configuration settings (SK_SEP7960A000001) and the SIP settings (SIP_3000001) described in the following two tables respectively:

Category	Settings
Scenario Editor	The test scenario comprises the phones registration followed by a call establishment procedure, bidirectional media streaming and call tear down. On the call originating Skinny scenario channel, Skinny phone A registers with the CallManager and originates a call by using the Skinny Make Call script uncton. Bidirectional media is exchanged by using the Voice Session function. The last scenario channel procedure is the Skinny End Call function that terminates the call. On the call terminating SIP scenario channel, SIP phone B registers with the CallManager as an SIP phone, and then executes an SIP Receive Call procedure to answer the call. The Voice Session function performs bidirectional media streaming. The call is terminated by using the SIP EndCall Receive procedure for the call terminating side.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time and to use one alias/channel.
Dial Plan	The Skinny registration names are defined by using a SEP7960A00[00001-] sequence generating expression. SIP_3000001:[5060] is configured as call destination.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skinny traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is selected. The industry-standard Express Forwarding (0xA0) TOS/DSCP setting for RTP traffic is selected.

Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skinny channels.
----------------	---

Category	Settings
Scenario Editor	The test scenario comprises the phones registration followed by a call establishment procedure, bidirectional media streaming and call tear down. On the call originating Skinny scenario channel, Skinny phone A registers with the CallManager and originates a call by using the Skinny Make Call script function. Bidirectional media is exchanged by using the Voice Session function. The last scenario channel procedure is the Skinny End Call function that terminates the call. On the call terminating SIP scenario channel, SIP phone B registers with the CallManager as an SIP phone, and then executes an SIP Receive Call procedure to answer the call. The Voice Session function performs bidirectional media streaming. The call is terminated by using the SIP EndCall Receive procedure for the call terminating side.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time and to use one alias/channel. The emulated SIP phones use consecutive IP addresses, the same port, and consecutive phone numbers.
Dial Plan	SIP endpoints phone numbers are defined by using a 30[00001-] sequence generating expression. Because this channel only terminates a call, no call destination needs to be configured.
SIP Settings	The Enable signaling on this activity option is selected for the SIP functions to be executed. The Port field in the SIP Settings area is specified by using a [5060-] sequence generating expression. In the Use Server area that needs to specify an SIP proxy server address and port, the address is that of the Cisco CallManager, and the port is 5060. Outbound proxy and registrar functionalities are configured.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is selected. In the Execution Settings page, the SIP phones are configured to use consecutive IPs and the same port. Hence, the RTP Port field is specified by using a single 10000 value.
Other Settings	The IP version preference is set to IPv4, and the following global scenario variables are initialized and can be used by all SIP channels: <ul style="list-style-type: none"> VoIP_Var0 is set to a 7960BBBB[0000-] sequence generating expression. This variable is used by the SIP Make Registration procedure in the SIP Send Request (INVITE) script function to register the SIP phone with a Cisco CallManager. VOIP_Var4 is set to '1.' This variable is used by the SIP Make Registration procedure to test whether the registration is executed for every loop (for a value of '1,') or only for the first loop (for a value of '2.')

MIX_018_7960_SM_US_900_ChS_IPv4_Static_SK_to_SIP_Call_Voice_3min

This test is similar to the previous one, with the only difference that the voice session has a duration of three minutes. The longer call duration is obtained by playing the selected wave file continuously for three minutes (**Advanced Playback Settings** tab of the Voice Session script function).

MIX_019_7960_SM_US_900_ChS_IPv4_Static_SK_to_SIP_Call_Voice_30_min

This test is similar to the previous one, with the only difference that the voice session has a duration of 30 minutes. The longer call duration is obtained by playing the selected wave file continuously for 30 minutes (**Advanced Playback Settings** tab of the Voice Session script function).

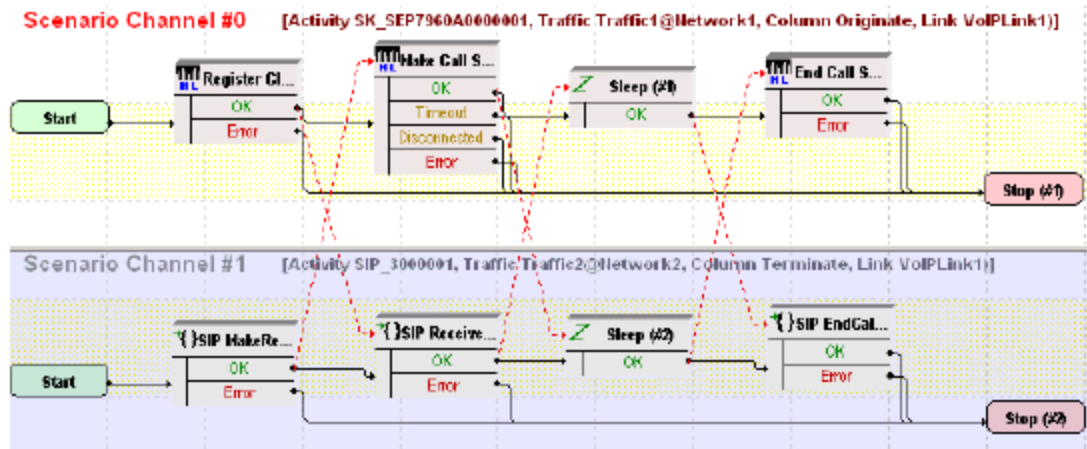
MIX_031_7960_SO_US_75k_BHCA_IPv4_Static_Sk_to_SIP_Call

This test illustrates a signaling-only mixed Skinny to SIP call with a configured objective of 75000 calls/hour that is to be attained using 3000 channels. The Talk Time parameter is computed automatically based on the values of the BHCA value and the specified number of channels.

The 7960 Skinny phones emulated by the SK_SEP7960A0000001 activity register with the Cisco CallManager, and then establish calls with the SIP phones emulated by the SIP_3000001 activity and registered with the CallManager as SIP phones. After the call is established, it is kept active for the duration of the computed Talk Time parameter by configuring the Sleep function by using the \$Talk-Time variable.

The call procedure is executed once for the entire test sustain time duration.

The underlying two-channel test scenario involving Skinny phone A and SIP phone B is shown in the following image:



The Skinny configuration settings (SK_SEP7960A0000001) and the SIP settings (SIP_3000001) are described in the following two tables respectively:

Category	Settings
----------	----------


Scenario Editor	The test scenario comprises the phones registration followed by a call establishment procedure, a call sustain time without media streaming, and call tear down. On the call originating Skinny scenario channel, Skinny phone A registers with the Cisco CallManager and originates a call by using the Skinny Make Call script function. In the Sleep script function, the call duration is set to the value of the Talk Time parameter. The last scenario channel function is the Skinny End Call function that terminates the call. On the call terminating SIP scenario channel, SIP phone B registers with the CallManager as an SIP phone, and then executes an SIP Receive Call procedure to answer the call. The Sleep function is configured with a duration equal to the Talk Time parameter. The call is terminated by using the SIP EndCall Receive procedure for the call terminating side.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time and to use one alias/channel.
Dial Plan	The Skinny registration names are defined by using a SEP7960A00[00001-] sequence generating expression. SIP_3000001:[5060] is configured as call destination.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skinny traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skinny channels.

Category	Settings
Scenario Editor	The test scenario comprises the phones registration followed by a call establishment procedure, a call sustain time without media streaming, and call tear down. On the call originating Skinny scenario channel, Skinny phone A registers with the Cisco CallManager and originates a call by using the Skinny Make Call script function. In the Sleep script function, the call duration is set to the value of the Talk Time parameter. The last scenario channel function is the Skinny End Call function that terminates the call. On the call terminating SIP scenario channel, SIP phone B registers with the CallManager as an SIP phone, and then executes an SIP Receive Call procedure to answer the call. The Sleep function is configured with a duration equal to the Talk Time parameter. The call is terminated by using the SIP EndCall Receive procedure for the call terminating side.

Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time. The emulated SIP phones use consecutive IP addresses, the same port, and consecutive phone numbers.
Dial Plan	SIP endpoints phone numbers are defined by using a 30[00001-] sequence generating expression. Because this channel only terminates a call, no call destination needs to be configured.
SIP Settings	The Enable signaling on this activity option is selected for the SIP functions to be executed. The Port field in the SIP Settings area is specified by using a [5060-] sequence generating expression. In the Use Server area that needs to specify an SIP proxy server address and port, the address is that of the Cisco CallManager, and the port is 5060. Outbound proxy and registrar functionalities are configured. The industry-standard Class 3 (0x60) TOS/DSCP setting for SIP traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and the following global scenario variables are initialized and can be used by all SIP channels: <ul style="list-style-type: none"> • VoIP_Var0 is set to a 7960BBBB[0000-] sequence generating expression. This variable is used by the SIP Make Registration procedure in the SIP Send Request (INVITE) script function to register the SIP phone with a Cisco CallManager. • VOIP_Var4 is set to '1.' This variable is used by the SIP Make Registration procedure to test whether the registration is executed for every loop (for a value of '1,') or only for the first loop (for a value of '2.')

MIX_029_7960_SM_US_75k_BHCA_IPv4_Static_Sk_to_SIP_Call_Voice

The test is similar to the previous one, with the difference that the underlying test scenario uses Voice Session script functions instead of Sleep function. Both Voice Session functions are configured to perform media streaming for the duration of the Talk Time parameter in the **Listen** and **Advanced Playback Settings** pages.

 **Note:** Because media streaming is performed, both the Skinny and the SIP activity need to have the **Enable media on this activity** option selected. On the VoIPSIP activity, the RTP port is defined by using a single 10000 value.

MIX_032_7960_SO_US_75k_BHCA_IPv4_Static_SIP_to_Sk_Call

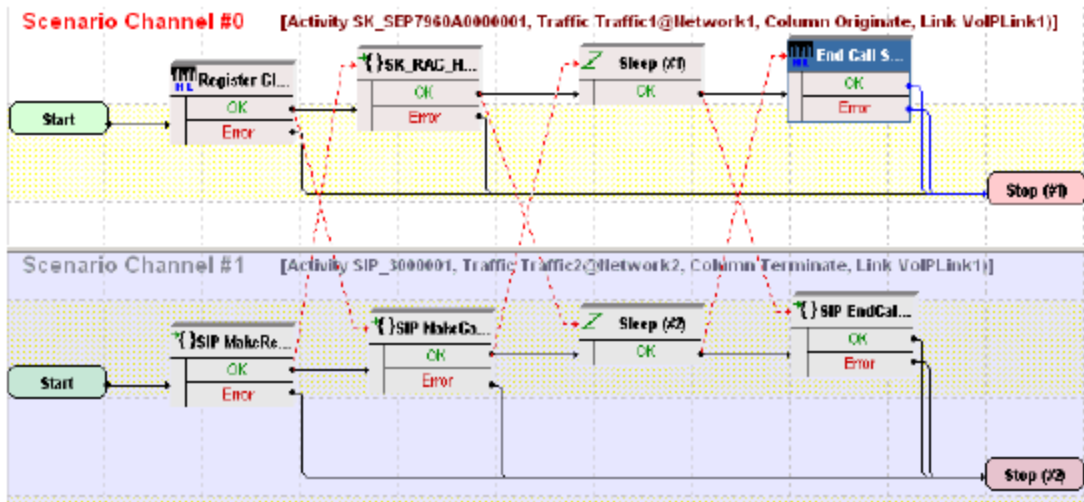
This test illustrates a mixed SIP to Skinny call with a configured objective of 75000 calls/hour that are to be attained using 3000 channels. The Talk Time parameter is computed automatically based on the values of the BHCA value and the number of channels.

The 7960 Skinny phones emulated by the SK_SEP7960A0000001 activity register with the Cisco CallManager and receive the calls originated by the SIP phones emulated by the SIP_3000001 activity

and registered with the Cisco Call-Manager as SIP phones. After the call is established, it is kept active for the duration of the computed Talk Time parameter by configuring the Sleep function by using the \$TalkTime variable.

The call procedure is executed once for the sustain time duration.

The underlying two-channel test scenario involving Skinny phone A and SIP phone B is shown in the following image:



To be able to run the test on your computer, select the Skinny configuration settings (SK_SEP7960A000001) and the SIP settings (SIP_3000001) described in the following two tables respectively:

Category	Settings
Scenario Editor	The test scenario comprises the phones registration followed by a call establishment procedure, a call sustain time without media streaming, and call tear down. On the call terminating Skinny scenario channel, Skinny phone A registers with the CallManager and receives the call by using the Sk_Receive AnswerCall_HL procedure. In the Sleep script function, the call duration is set to the value of the Talk Time parameter. The last scenario channel function is the Skinny End Call function that terminates the call. On the call originating SIP scenario channel, SIP phone B registers with the CallManager as an SIP phone, and then executes an SIP Make Call procedure. The Sleep function is configured with a duration equal to the Talk Time parameter. The call is terminated by using the SIP EndCall Receive procedure for the call terminating side.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time.
Dial Plan	The Skinny registration names are defined by using a SEP7960A00[00001-] sequence generating expression. Because the channel only terminates a call, no call destination needs to be configured.


Skippy Settings	The Enable signaling on this activity option is selected for the Skippy functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skippy traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skippy channels.

Category	Settings
Scenario Editor	The test scenario comprises the phones registration followed by a call establishment procedure, a call sustain time without media streaming, and call tear down. On the call terminating Skippy scenario channel, Skippy phone A registers with the CallManager and receives the call by using the Sk_Receive AnswerCall_HL procedure. In the Sleep script function, the call duration is set to the value of the Talk Time parameter. The last scenario channel function is the Skippy End Call function that terminates the call. On the call originating SIP scenario channel, SIP phone B registers with the CallManager as an SIP phone, and then executes an SIP Make Call procedure. The Sleep function is configured with a duration equal to the Talk Time parameter. The call is terminated by using the SIP EndCall Receive procedure for the call terminating side.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time and to use one alias/channel. The emulated SIP phones use consecutive IP address, the same port, and consecutive phone numbers.
Dial Plan	SIP endpoints phone numbers are defined by using a 30[00001-] sequence generating expression. SK_SEP7960A0000001 is configured as call destination.
SIP Settings	The Enable signaling on this activity option is selected for the SIP functions to be executed. The Port field in the SIP Settings area is specified by using a [5060-] sequence generating expression. In the Use Server area that needs to specify an SIP proxy server address and port, the address is that of the Cisco CallManager, and the port is 5060. Outbound proxy and registrar functionalities are configured. The industry-standard Class 3 (0x60) TOS/DSCP setting for SIP traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.

Other Settings	<p>The IP version preference is set to IPv4, and the following global scenario variables are initialized and can be used by all SIP channels:</p> <ul style="list-style-type: none">• VoIP_Var0 is set to a 7960BBBB[0000-] sequence generating expression. This variable is used by the SIP Make Registration procedure in the SIP Send Request (INVITE) script function to register the SIP phone with a Cisco CallManager.• VOIP_Var4 is set to '1.' This variable is used by the SIP Make Registration procedure to test whether the registration is executed for every loop (for a value of '1,') or only for the first loop (for a value of '2.')
----------------	--

MIX_030_7960_SM_US_75k_BHCA_IPv4_Static_SIP_to_Sk_Call_Voice

The test is similar to the previous one, with the difference that the underlying test scenario uses Voice Session script functions instead of Sleep function. Both Voice Session functions are configured to perform media streaming for the duration of the Talk Time parameter in the **Listen** and **Advanced Playback Settings** pages.

 **Note:** Because media streaming is performed during the test, both the VoIPSkinny and VoIPSIP activity need to have the **Enable media on this activity** option selected. On the VoIPSIP activity, the RTP port is configured by using a single 10000 value.

Mixed Skinny and SIP - SIP Trunk


This group of sample configuration files are used for testing the SIP trunk support implemented by the Cisco CallManager 4.x.

Sample configurations, used for both signaling-only and signaling with media tests, comprise emulated Skinny IP phones registered with a Cisco CallManager that establish calls to SIP phones through an SIP trunk.

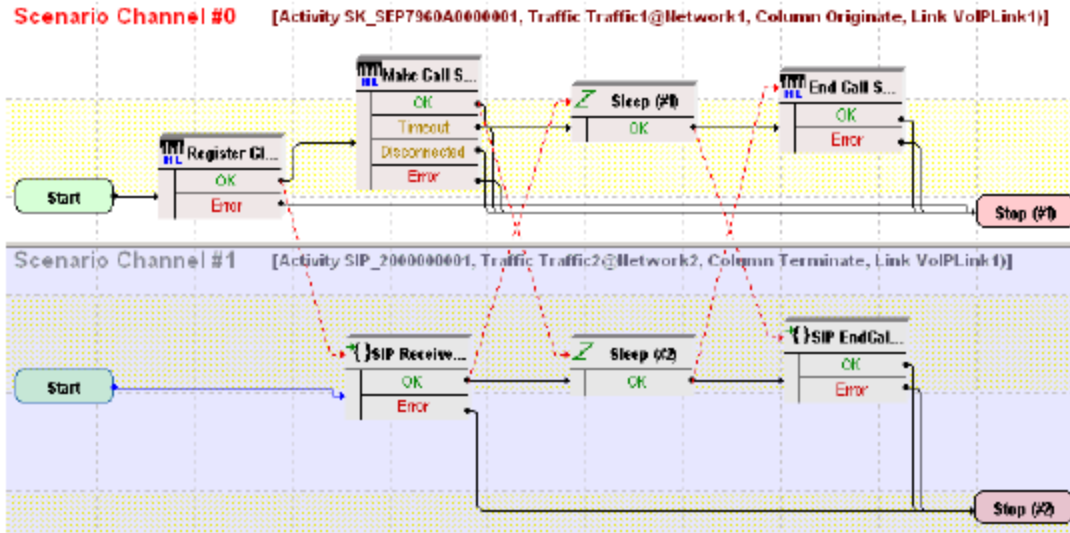
MIX_001_7960_SO_US_5k_ChS_IPv4_Static_SK_to_SIP_trunk_Bulk_Call_10s

This test illustrates a mixed Skinny to SIP call procedure without media streaming. The Skinny phones emulated by the SK_SEP7960A0000001 activity register with the Cisco CallManager, and then establish calls with the SIP phones – emulated by the SIP_2000000001 activity – through an SIP trunk. After the call is established, the connection is kept up for the duration of the Sleep functions, configured to 10 seconds.

The call procedure runs once for the test duration.

 **Important!** This test assumes that an SIP trunk has been configured between the Cisco CallManager and IxLoad, with the trunk configuration having the Significant Digits option configured to the **All** value.

The underlying two-channel test scenario involving Skinny phone A and SIP phone B is shown in the following image:



SK_SEP7960A0000001 and SIP_2000000001 configured settings are described in the following two tables respectively:

Category	Settings
Scenario Editor	On the call originating Skinny scenario channel, phone A registers with the Cisco CallManager and originates a call to phone B through an SIP trunk configured on the CallManager. After call initiation by using the Skinny Make Call script function, the call duration is configured to 10 seconds by using the Sleep function. Finally, phone A executes the Skinny End Call function that terminates the call. On the call terminating SIP scenario channel, SIP phone B executes an SIP Receive Call procedure to answer the call. The Sleep function ensures a call duration of 10 seconds. Eventually, the SIP EndCall Receive procedure is used for the call terminating side.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time.
Dial Plan	The Skinny registration names are defined by using a SEP7960A00[00001-] sequence generating expression. SIP_2000000001:5060 is configured as call destination.
Skiny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skinny traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.

Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skinny channels.
----------------	---

Category	Settings
Scenario Editor	On the call originating Skinny scenario channel, phone A registers with the Cisco CallManager and originates a call to phone B through an SIP trunk configured on the CallManager. After call initiation by using the Skinny Make Call script function, the call duration is configured to 10 seconds by using the Sleep function. Finally, phone A executes the Skinny End Call function that terminates the call. On the call terminating SIP scenario channel, SIP phone B executes an SIP Receive Call procedure to answer the call. The Sleep function ensures a call duration of 10 seconds. Eventually, the SIP EndCall Receive procedure is used for the call terminating side.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time. The emulated SIP phones use consecutive IP address, the same port, and consecutive phone numbers. The Accept multiple channels sharing the same IP:port option is selected.
Dial Plan	SIP phone numbers are defined by using a 20000[00001-] sequence generating expression. Because this channel only terminates a call, no call destination needs to be configured.
SIP Settings	The Enable signaling on this activity option is selected for the SIP functions to be executed. The Port field in the SIP Settings area is specified by using a single 5060 value. The industry-standard Class 3 (0x60) TOS/DSCP setting for SIP traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the SIP channels.

MIX_002_7960_SO_US_5k_ChS_IPv4_Static_SK_to_SIP_trunk_Bulk_Call_3min

This test is similar to the previous one, with the only difference that the call duration is three minutes, configured by using the Sleep script function.

MIX_003_7960_SO_US_5k_ChS_IPv4_Static_SK_to_SIP_trunk_Bulk_Call_30min

This test is similar to the previous one, with the only difference that the call duration is 30 minutes, configured by using the Sleep script function.

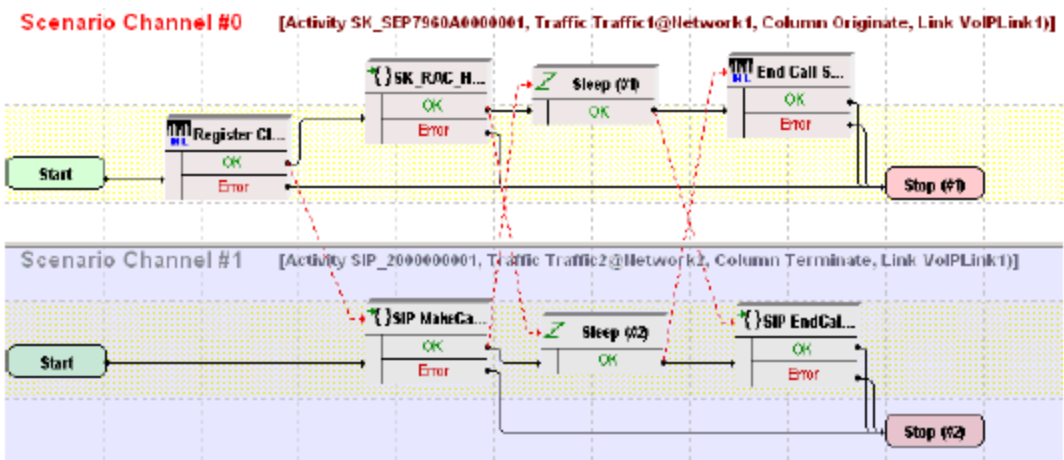
MIX_004_7960_SO_US_5k_ChS_IPv4_Static_SIP_to_SK_trunk_Bulk_Call_10s

This test illustrates a mixed SIP to Skinny call procedure without media streaming. SIP phones emulated by the SIP_2000000001 activity establish calls through an SIP trunk to a Cisco CallManager, with whom the Skinny phones emulated by the SK_SEP7960A0000001 activity are registered. After call establishment, a call duration of 10 seconds is configured by using Sleep script functions on both scenario channels.

The call procedure is repeated once for the test duration.

! Important! This test assumes that an SIP trunk has been configured between the Cisco CallManager and IxLoad, with the trunk configuration having the **Significant Digits** option configured to the **All** value.

The underlying two-channel test scenario involving Skinny phone A and SIP phone B is shown in the following image:



SK_SEP7960A0000001 and SIP_2000000001 configured settings are described in the following two tables respectively:

Category	Settings
Scenario Editor	On the call originating scenario channel, SIP phone B originates a call to Skinny phone A through an SIP trunk configured on the CallManager. After call initiation by using the SIP MakeCall procedure, the call is maintained active for a duration of 10 seconds by using the Sleep script function. The last SIP EndCall Receive procedure is a call termination procedure for the receiving side. On the call terminating Skinny channel, Skinny phone A registers with the Cisco CallManager and executes an Sk_Receive AnswerCall_HL procedure – a wrapping of the common Wait Call and Answer Call script functions – to answer the call. The call is maintained active for a duration of 10 seconds by using the Sleep script function, before it is terminated by using the Skinny End Call script function.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time.

Dial Plan	The Skinny phone numbers are specified by using a 16[00001-] sequence generation expression. The Skinny registration names are defined by using a SEP7960A00[00001-] sequence generating expression. Because this activity only terminates a call, no call destination and no transfer destination needs to be configured.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skinny traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skinny channels.

Category	Settings
Scenario Editor	On the call originating scenario channel, SIP phone B originates a call to Skinny phone A through an SIP trunk configured on the CallManager. After call initiation by using the SIP MakeCall procedure, the call is maintained active for a duration of 10 seconds by using the Sleep script function. The last SIP EndCall Receive procedure is a call termination procedure for the receiving side. On the call terminating Skinny channel, Skinny phone A registers with the Cisco CallManager and executes an Sk_Receive AnswerCall_HL procedure – a wrapping of the common Wait Call and Answer Call script functions – to answer the call. The call is maintained active for a duration of 10 seconds by using the Sleep script function, before it is terminated by using the Skinny End Call script function.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time. The emulated SIP phones use consecutive IP address, the same port, and consecutive phone numbers. The Accept multiple channels sharing the same IP:port option is selected.
Dial Plan	SIP phone numbers are defined by using a 20000[00001-] sequence generating expression. The configured call destination is the address of the CallManager and the 5080 port. The override numbers of destination activity option is selected and the Skinny phone numbers are specified by using a 16[00001-] sequence generation expression.

SIP Settings	The Enable signaling on this activity option is selected for the SIP functions to be executed. The Port field in the SIP Settings area is specified by using a single 5060 value. The industry-standard Class 3 (0x60) TOS/DSCP setting for SIP traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the SIP channels.

MIX_005_7960_SO_US_5k_ChS_IPv4_Static_SIP_to_SK_trunk_Bulk_Call_3min

This test is similar to the previous one, with the only difference that the call duration is three minutes, configured by using the Sleep script function.

MIX_006_7960_SO_US_5k_ChS_IPv4_Static_SIP_to_SK_trunk_Bulk_Call_30min

This test is similar to the previous one, with the only difference that the call duration is 30 minutes, configured by using the Sleep script function.

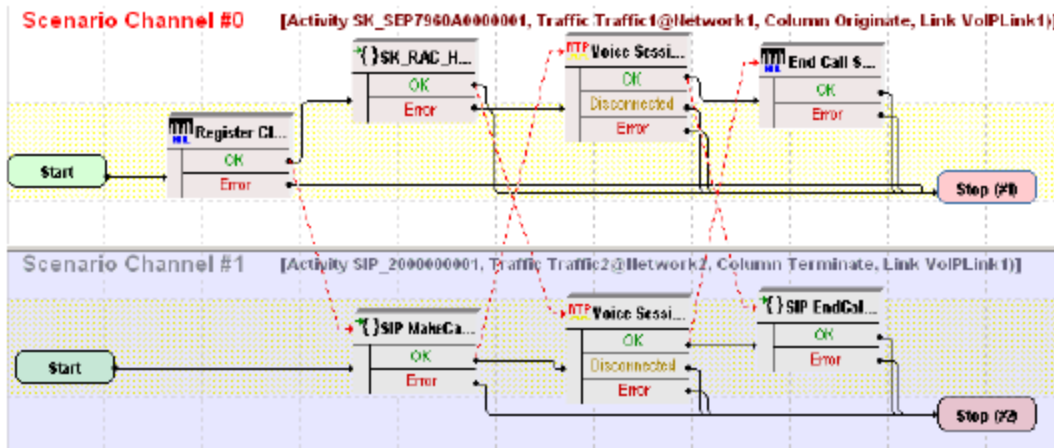
MIX_012_7960_SM_US_900_ChS_IPv4_Static_SIP_to_SK_trunk_Call_Voice_10s

This test illustrates a mixed SIP to Skinny call scenario with media streaming. SIP phones emulated by the SIP_2000000001 activity establish calls through an SIP trunk to a Cisco CallManager, with whom the Skinny phones emulated by the SK_SEP7960A0000001 activity are registered. After establishing the call, the phones use the Voice Session script function for bidirectional media streaming.

! **Important!** This test assumes that an SIP trunk has been configured between the Cisco CallManager and IxLoad, with the trunk configuration having the **Significant Digits** option configured to the **All** value.

The call scenario runs once for the duration of the test.


The underlying two-channel test scenario involving Skinny phone A and SIP phone B is shown in the following image:



SK_SEP7960A0000001 and SIP_2000000001 configured settings are described in the following two tables respectively:

Category	Settings
Scenario Editor	On the call originating scenario channel, SIP phone B originates a call to Skinny phone A through an SIP trunk configured on the CallManager. After call establishment by using the SIP MakeCall procedure, media is exchanged by using the Voice Session script function. The last procedure executed by SIP phone B is a call termination procedure for the receiving side. On the call terminating scenario channel, Skinny phone A registers with the CallManager and executes an Sk_Receive AnswerCall_HL procedure – a wrapping of the common Wait Call and Answer Call script functions – to answer the call. The Voice Session function is used to perform bidirectional media streaming. Finally, phone A executes the Skinny End Call script function to tear down the call.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time.
Dial Plan	The Skinny registration names are defined by using the 7960AAAA[0000-] sequence generating expression. The Skinny phone numbers are specified by using a 16[00001-] sequence generation expression. Because this activity is terminating a call, no call destination needs to be configured.
Skippy Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skinny traffic is selected.
Codec Settings	The default codec settings are used.

RTP Settings	The Enable media on this activity option is selected, with RTP port 10000 specified. The industry-standard Express Forwarding (0xA0) TOS/DSCP setting for RTP traffic is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skinny channels.

Category	Settings
Scenario Editor	On the call originating scenario channel, SIP phone B originates a call to Skinny phone A through an SIP trunk configured on the CallManager. After call establishment by using the SIP MakeCall procedure, media is exchanged by using the Voice Session script function. The last procedure executed by SIP phone B is a call termination procedure for the receiving side. On the call terminating scenario channel, Skinny phone A registers with the CallManager and executes an Sk_Receive AnswerCall_HL procedure – a wrapping of the common Wait Call and Answer Call script functions – to answer the call. The Voice Session function is used to perform bidirectional media streaming. Finally, phone A executes the Skinny End Call script function to tear down the call.
Execution Settings	The corresponding scenario channel is configured to execute once during the test duration. The emulated SIP phones use the same IP address and port, and consecutive phone numbers. The Accept multiple channels sharing the same IP:port option is selected.  Note: The Use consecutive value setting for the TCP/UDP port is intended for correct configuration of RTP ports.
Dial Plan	SIP phone numbers are defined by using a 20000[00001-] sequence generating expression. The address and port 5080 of the Cisco CallManager are configured as call destination. The override numbers of destination activity option is selected and the Skinny phone numbers are specified by using a 16[00001-] sequence generation expression.
SIP Settings	The Enable signaling on this activity option is selected for the SIP functions to be executed. The Port field in the SIP Settings area is specified by using a single 5060 value. The industry-standard Class 3 (0x60) TOS/DSCP setting for SIP traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is selected. The RTP Port field is specified by using a [10000-2000,2] sequence generating expression.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the SIP channels.

MIX_013_7960_SM_US_900_ChS_IPv4_Static_SIP_SK_trunk_Call_Voice_3min

This test is similar to the previous one, with the only difference that the voice session has a duration of three minutes. The longer call duration is obtained by playing the selected wave file continuously for three minutes (**Advanced Playback Settings** tab of the Voice Session script function).

MIX_014_7960_SM_US_900_ChS_IPv4_Static_SIP_SK_trunk_Call_Voice_30min

This test is similar to the previous one, with the only difference that the voice session has a duration of 30 minutes. The longer call duration is obtained by playing the selected wave file continuously for 30 minutes (**Advanced Playback Settings** tab of the Voice Session script function).

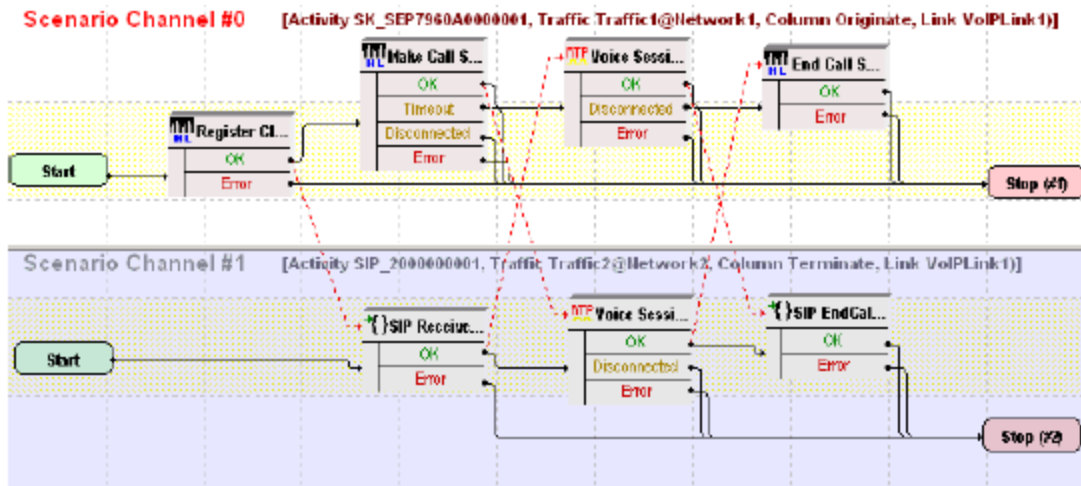
MIX_009_7960_SM_US_900_ChS_IPv4_Static_SK_to_SIP_trunk_Call_Voice_10s

This test illustrates a mixed Skinny to SIP call scenario with media streaming. A number of Skinny phones emulated by the SK_SEP7960A0000001 activity register with the Cisco CallManager and then originate calls to the SIP phones, emulated by the SIP_2000000001 activity, through an SIP trunk. After establishing the call, the phones use the Voice Session script function for bidirectional media streaming.

The call scenario runs once for the duration of the test.

! Important! This test assumes that an SIP trunk has been configured between the Cisco CallManager and IxLoad, with the trunk configuration having the **Significant Digits** option configured to the **All** value.

The underlying two-channel test scenario involving Skinny phone A and SIP Trunk B is shown in the following image:




SK_SEP7960A0000001 and SIP_2000000001 configured settings are described in the following two tables respectively:

Category	Settings
----------	----------

Scenario Editor	On the call originating scenario channel, Skinny phone A, registered with the CallManager, originates a call through an SIP trunk configured on the CallManager to SIP phone B. After call establishment by using the Skinny Make Call script function, bidirectional media is exchanged. The last scenario channel procedure executed by phone A is the Skinny End Call function that terminates the call. On the call terminating scenario channel, SIP phone B executes an SIP Receive Call procedure to answer the call. The Voice Session function performs bidirectional media streaming. Finally, phone B executes the SIP EndCall Receive procedure for the call terminating side.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time.
Dial Plan	The Skinny registration names are defined by using the 7960AAAA[0000-] sequence generating expression. The Skinny phone numbers are specified by using a 16[00001-] sequence generation expression. SIP_2000000001:5060 is configured as call destination.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skinny traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is selected, with RTP port 10000 specified. The industry-standard Express Forwarding (0xA0) TOS/DSCP setting for RTP traffic is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skinny channels.

Category	Settings
Scenario Editor	On the call originating scenario channel, Skinny phone A, registered with the CallManager, originates a call through an SIP trunk configured on the CallManager to SIP phone B. After call establishment by using the Skinny Make Call script function, bidirectional media is exchanged. The last scenario channel procedure executed by phone A is the Skinny End Call function that terminates the call. On the call terminating scenario channel, SIP phone B executes an SIP Receive Call procedure to answer the call. The Voice Session function performs bidirectional media streaming. Finally, phone B executes the SIP EndCall Receive procedure for the call terminating side.

Execution Settings	<p>The corresponding scenario channel is configured to execute once during the test duration. The emulated SIP phones use the same IP address and port, and consecutive phone numbers. The Accept multiple channels sharing the same IP:port option is selected.</p> <hr/> <p> Note: The Use consecutive value setting for the TCP/UDP port is intended for correct configuration of RTP ports.</p>
Dial Plan	SIP phone numbers are defined by using a 20000[00001-] sequence generating expression. Because this activity only terminates a call, no call destination needs to be configured.
SIP Settings	The Enable signaling on this activity option is selected for the SIP functions to be executed. The Port field in the SIP Settings area is specified by using a single 5060 value.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is selected. The RTP Port field is specified by using a [10000-2000,2] sequence generating expression.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the SIP channels.

MIX_010_7960_SM_US_900_ChS_IPv4_Static_SK_to_SIP_trunk_Call_Voice_3min

This test is similar to the previous one, with the only difference that the voice session has a duration of three minutes. The longer call duration is obtained by playing the selected wave file continuously for three minutes (**Advanced Playback Settings** tab of the Voice Session script function).

MIX_011_7960_SM_US_900_ChS_IPv4_Static_SK_to_SIP_trunk_Call_Voice_30min

This test is similar to the previous one, with the only difference that the voice session has a duration of 30 minutes. The longer call duration is obtained by playing the selected wave file continuously for 30 minutes (**Advanced Playback Settings** tab of the Voice Session script function).

MIX_008_7960_SO_US_80k_BHCA_IPv4_Static_SIP_to_SK_trunk_Bulk_Call

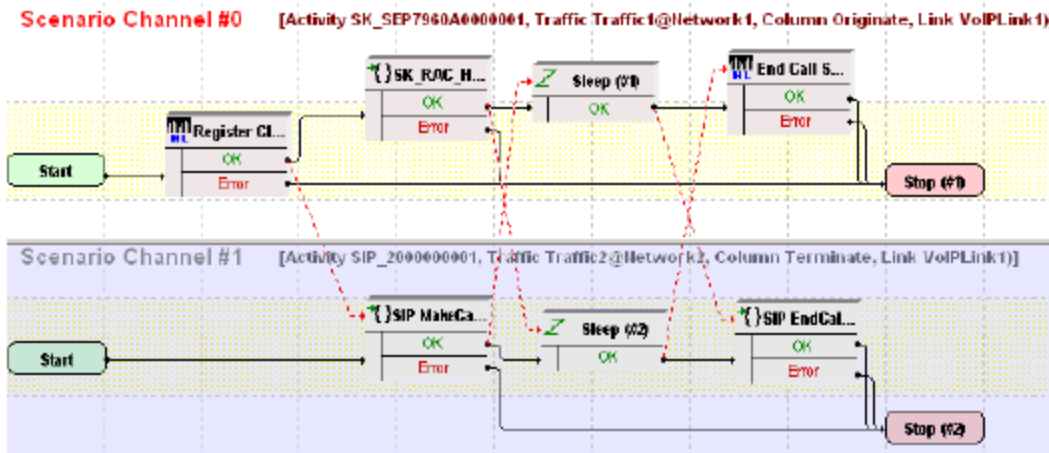
This test illustrates a mixed SIP to Skinny call procedure without media streaming, having a configured objective of 80000 calls/hour that is to be attained by using 5000 channels. The Talk Time parameter is computed automatically based on the values of the BHCA value and the specified number of channels.

The SIP phones emulated by the SIP_2000000001 activity establish calls through an SIP trunk to a Cisco CallManager, with whom the Skinny phones emulated by the SK_SEP7960A0000001 activity are registered. After the call is established, it is kept active for the duration of the computed Talk Time parameter by configuring the Sleep function by using the \$TalkTime variable.

Important! This test assumes that an SIP trunk has been configured between the Cisco CallManager and IxLoad, with the trunk configuration having the **Significant Digits** option configured to the **All** value.

The test is run once for the sustain time duration.

The underlying two-channel test scenario involving Skinny phone A and SIP phone B is shown in the following image:



SK_SEP7960A0000001 and SIP_2000000001 configured settings are described in the following two tables respectively:

Category	Settings
Scenario Editor	On the call originating scenario channel, SIP phone B originates a call to Skinny phone A through an SIP trunk configured on the CallManager. After call initiation by using the SIP MakeCall procedure, the Sleep script function configures the call duration to the value of the Talk Time parameter. The last procedure executed by SIP phone B is a call termination procedure for the receiving side. On the call terminating scenario channel, Skinny phone A registers with the CallManager and executes a Sk_Receive AnswerCall_HL procedure – a wrapping of the common Wait Call and Answer Call script functions – to answer the call. In the Sleep script function, the call duration is set to the value of the Talk Time parameter. Finally, phone A executes the Skinny End Call script function to tear down the call.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time.
Dial Plan	The Skinny registration names are defined by using the SEP7960A00[00001-] sequence generating expression. The Skinny phone numbers are specified by using a 16[00001-] sequence generation expression. Because the channel only terminates a call, no call destination needs to be configured.

Skinnny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skinny traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skinny channels.

Category	Settings
Scenario Editor	On the call originating scenario channel, SIP phone B originates a call to Skinny phone A through an SIP trunk configured on the CallManager. After call initiation by using the SIP MakeCall procedure, the Sleep script function configures the call duration to the value of the Talk Time parameter. The last procedure executed by SIP phone B is a call termination procedure for the receiving side. On the call terminating scenario channel, Skinny phone A registers with the CallManager and executes a Sk_Receive AnswerCall_HL procedure – a wrapping of the common Wait Call and Answer Call script functions – to answer the call. In the Sleep script function, the call duration is set to the value of the Talk Time parameter. Finally, phone A executes the Skinny End Call script function to tear down the call.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time. The emulated SIP phones use the same IP address and port, and consecutive phone numbers. The Accept multiple channels sharing the same IP:port option is selected.
Dial Plan	SIP phone numbers are defined by using a 20000[00001-] sequence generating expression. The address of the Cisco CallManager and port 5080 are configured as call destination. The override numbers of destination activity option is selected and the Skinny phone numbers are specified by using a 16[00001-] sequence generation expression.
SIP Settings	The Enable signaling on this activity option is selected for the SIP functions to be executed. The Port field in the SIP Settings area is specified by using a single 5060 value. The industry-standard Class 3 (0x60) TOS/DSCP setting for SIP traffic is selected.
Codec Settings	The default codec settings are used.

RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the SIP channels.

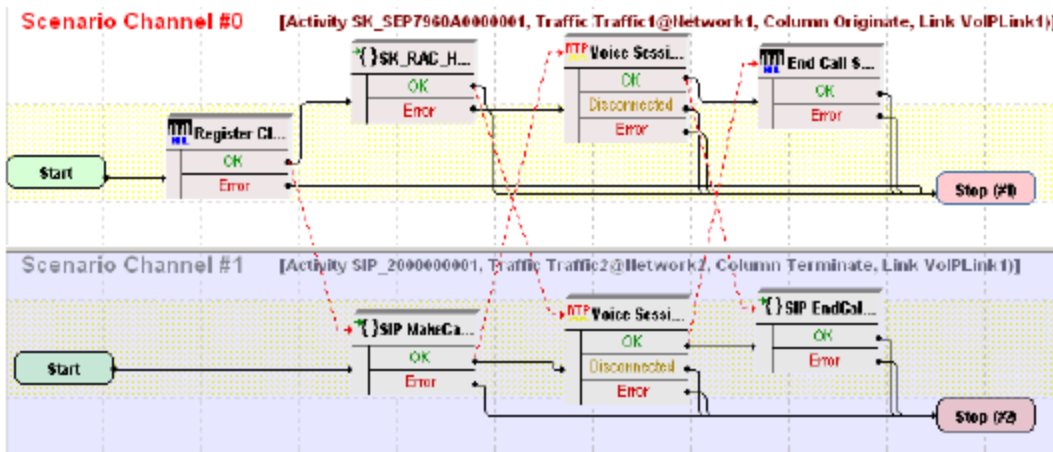
MIX_016_7960_SM_US_75k_BHCA_IPv4_Static_SIP_to_SK_trunk_Call_Voice

This test illustrates a mixed SIP to Skinny call procedure with media streaming. SIP phones emulated by the SIP_2000000001 activity establish calls through an SIP trunk to a Cisco CallManager, with whom the Skinny phones emulated by the SK_SEP7960A0000001 activity are registered. After establishing the call, the phones use the Voice Session script function for bidirectional media streaming. Both Voice Session functions are configured to perform media streaming for the duration of the Talk Time parameter in the **Listen** and **Advanced Playback Settings** function pages.

! Important! This test assumes that an SIP trunk has been configured between the Cisco CallManager and IxLoad, with the trunk configuration having the **Significant Digits** option configured to the **All** value.

The call procedure is repeated once for the test duration.

The underlying two-channel test scenario involving Skinny phone A and SIP phone B is shown in the following image:




SK_SEP7960A0000001 and SIP_2000000001 configured settings are described in the following two tables respectively:

Category	Settings
----------	----------

Scenario Editor	On the call originating scenario channel, SIP phone B originates a call to Skinny phone A through an SIP trunk configured on the CallManager. After call establishment by using the SIP MakeCall procedure, media is exchanged by using the Voice Session script function. The last procedure executed by SIP phone B is a call termination procedure for the receiving side. On the call terminating scenario channel, Skinny phone A registers with the CallManager and executes an Sk_Receive AnswerCall_HL procedure – a wrapping of the common Wait Call and Answer Call script functions – to answer the call. The Voice Session function is used to perform bidirectional media streaming. Finally, phone A executes the Skinny End Call script function to tear down the call.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time.
Dial Plan	The Skinny registration names are defined by using the SEP7960A00[00001-] sequence generating expression. The Skinny phone numbers are specified by using a 16[00001-] sequence generation expression. No call destination is configured.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skinny traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is selected and RTP port 10000 is specified. The industry-standard Express Forwarding (0xA0) TOS/DSCP setting for RTP traffic is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skinny channels.


Category	Settings
Scenario Editor	On the call originating scenario channel, SIP phone B originates a call to Skinny phone A through an SIP trunk configured on the CallManager. After call establishment by using the SIP MakeCall procedure, media is exchanged by using the Voice Session script function. The last procedure executed by SIP phone B is a call termination procedure for the receiving side. On the call terminating scenario channel, Skinny phone A registers with the CallManager and executes an Sk_Receive AnswerCall_HL procedure – a wrapping of the common Wait Call and Answer Call script functions – to answer the call. The Voice Session function is used to perform bidirectional media streaming. Finally, phone A executes the Skinny End Call script function to tear down the call.

Execution Settings	<p>The corresponding scenario channel is configured to execute once during the test sustain time. The emulated SIP phones use the same IP address and port, and consecutive phone numbers. The Accept multiple channels sharing the same IP:port option is selected.</p> <hr/> <p> Note: The Use consecutive value setting for the TCP/UDP port is intended for correct configuration of RTP ports.</p>
Dial Plan	<p>SIP phone numbers are defined by using a 20000[00001-] sequence generating expression. The address of the Cisco CallManager and port 5080 are configured as call destination. The override numbers of destination activity option is selected and the Skinny phone numbers are specified by using a 16[00001-] sequence generation expression.</p>
SIP Settings	<p>The Enable signaling on this activity option is selected for the SIP functions to be executed. The Port field in the SIP Settings area is specified by using a single 5060 value.</p>
Codec Settings	<p>The default codec settings are used.</p>
RTP Settings	<p>The Enable media on this activity option is selected. The RTP Port field is specified by using a [10000-2000,2] sequence generating expression.</p>
Other Settings	<p>The IP version preference is set to IPv4, and no scenario variables need to be initialized for the SIP channels.</p>

MIX_007_7960_SO_US_80k_BHCA_IPv4_Static_SK_to_SIP_trunk_Bulk_Call

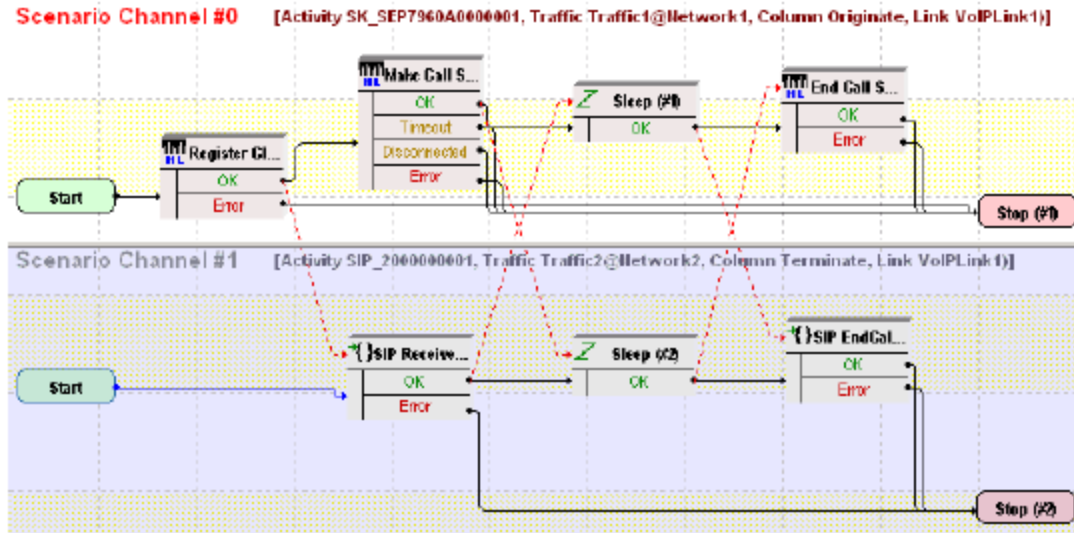
This test illustrates a mixed Skinny to SIP call procedure without media streaming, having a configured objective of 80000 calls/hour that is to be attained by using 3000 channels. The Talk Time parameter is computed automatically based on the values of the BHCA value and the number of channels.

The Skinny phones emulated by the SK_SEP7960A0000001 activity register with a Cisco CallManager, and then originate calls to the SIP phones, emulated by the SIP_2000000001 activity, through an SIP trunk. After the call is established, it is kept active for the duration of the computed Talk Time parameter by configuring the Sleep function by using the \$TalkTime variable.

 **Note:** This test assumes that an SIP trunk has been configured between the Cisco CallManager and IxLoad, with the trunk configuration having the **Significant Digits** option configured to the **All** value.

The call procedure is run once for the test duration.

The underlying two-channel test scenario involving Skinny phone A and SIP phone B is shown in the following image:



SK_SEP7960A0000001 and SIP_2000000001 configured settings are described in the following two tables respectively:

Category	Settings
Scenario Editor	On the call originating scenario channel, Skinny phone A registers with the CallManager and originates a call through the SIP trunk configured on the CallManager to phone B by using the Skinny Make Call script function. After call initiation by using the Skinny Make Call function, the Sleep function configures the call duration to the value of the Talk Time parameter. The last scenario channel procedure executed by phone A is the Skinny End Call function that terminates the call. On the call terminating scenario channel, phone B executes a SIP Receive Call procedure to answer the call. In the Sleep script function, the call duration is set to the value of the Talk Time parameter. Finally, phone B executes the SIP EndCall Receive procedure for the call terminating side.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time.
Dial Plan	The Skinny registration names are defined by using the SEP7960A00[00001-] sequence generating expression. SIP_2000000001:5060 is configured as call destination.
Skippy Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skinny traffic is selected.
Codec Settings	The default codec settings are used.

RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skinny channels.

Category	Settings
Scenario Editor	On the call originating scenario channel, Skinny phone A registers with the CallManager and originates a call through the SIP trunk configured on the CallManager to phone B by using the Skinny Make Call script function. After call initiation by using the Skinny Make Call function, the Sleep function configures the call duration to the value of the Talk Time parameter. The last scenario channel procedure executed by phone A is the Skinny End Call function that terminates the call. On the call terminating scenario channel, phone B executes a SIP Receive Call procedure to answer the call. In the Sleep script function, the call duration is set to the value of the Talk Time parameter. Finally, phone B executes the SIP EndCall Receive procedure for the call terminating side.
Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time. The emulated SIP phones use the same IP address and port, and consecutive phone numbers. The Accept multiple channels sharing the same IP:port option is selected.
Dial Plan	SIP phone numbers are defined by using a 20000[00001-] sequence generating expression. Because this channel only terminates a call, no call destination needs configured.
SIP Settings	The Enable signaling on this activity option is selected for the SIP functions to be executed. The Port field in the SIP Settings area is specified by using a single 5060 value. The industry-standard Class 3 (0x60) TOS/DSCP setting for SIP traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	Because this test does not perform media streaming, the Enable media on this activity option is not selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the SIP channels.

MIX_015_7960_SM_US_75k_BHCA_IPv4_Static_Sk_to_SIP_trunk_Call_Voice

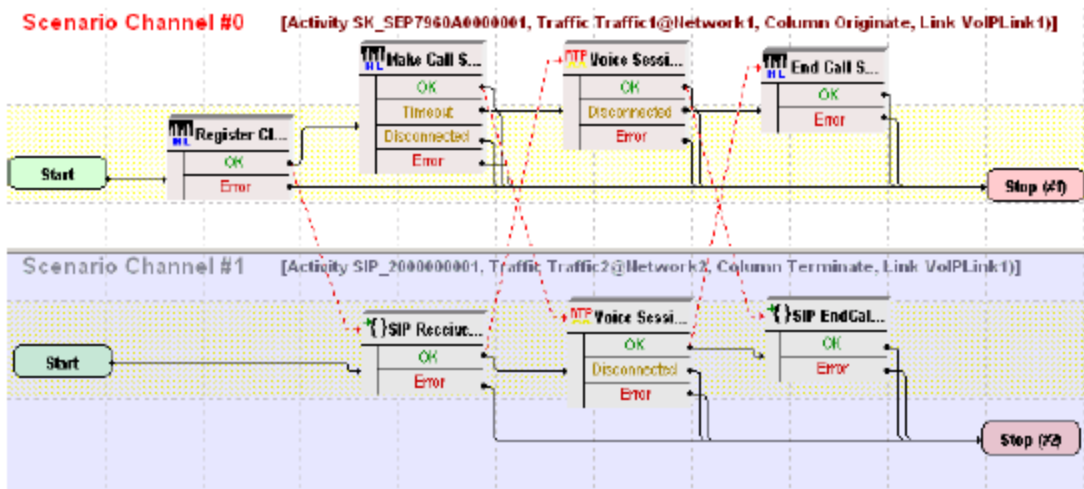
This test illustrates a mixed Skinny to call SIP procedure with media streaming.

The Skinny phones emulated by the SK_SEP7960A0000001 activity register with a Cisco CallManager, and then originate calls to the SIP phones, emulated by the SIP_2000000001 activity, through an SIP trunk. After establishing the call, the phones use the Voice Session script function for bidirectional media streaming. Both Voice Session functions are configured to perform media streaming for the duration of the Talk Time parameter in the **Listen** and **Advanced Playback Settings** script function pages.

The call procedure is repeated once for the test duration.

! Important! This test assumes that an SIP trunk has been configured between the Cisco CallManager and IxLoad, with the trunk configuration having the **Significant Digits** option configured to the **All** value.

The underlying two-channel test scenario involving Skinny phone A and SIP phone B is shown in the following image:



SK_SEP7960A0000001 and SIP_2000000001 configured settings are described in the following two tables respectively:


Category	Settings
Scenario Editor	On the call originating scenario channel, Skinny phone A registers with the CallManager and originates a call to SIP phone B through an SIP trunk configured on the CallManager. After call initiation by using the Skinny Make Call function, the Voice Session function is used to perform bidirectional media streaming for the duration of the Talk Time parameter. The last scenario channel procedure executed by phone A is the Skinny End Call function that terminates the call. On the call terminating scenario channel, phone B executes an SIP Receive Call procedure to answer the call. After call establishment, media is exchanged by using the Voice Session script function. Finally, phone B executes the SIP EndCall Receive procedure for the call terminating side.


Execution Settings	The corresponding scenario channel is configured to execute once during the test sustain time.
Dial Plan	The Skinny registration names are defined by using the 7960AAAA[0000-] sequence generating expression. SIP_2000000001:5060 is configured as call destination.
Skinny Settings	The Enable signaling on this activity option is selected for the Skinny functions to be executed. The Call Manager IP address and port need to be configured in the Call Managers area. The industry-standard Class 3 (0x60) TOS/DSCP setting for Skinny traffic is selected.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is selected and RTP port 10000 is specified. The industry-standard Express Forwarding (0xA0) TOS/DSCP setting for RTP traffic is selected.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the Skinny channels.

H.323 Sample Test Configuratings and Test Scenarios

Using the functions from the VoIP H.323 test library, you can generate and execute a large number of originator/answerer scenario configurations that comply with the H.323 protocol. You can use one of the predefined test scenarios described in this chapter or create a new one, map it to VoIP H.323 activities, and start the execution.

This section describes the pre-defined IxLoad Voice Plug-in H.323 available sample test configurations (RXFs) and their associated test scenarios.

 **Note:** For a complete description of the supported H.323 test library functions, see [VoIP H323 Functions Library](#).

 **Note:** Sample tests follow a naming convention that comprises the test type (VH for VoIP H323), an index, a test configuration (B2B for Back-to-Back or GK for running against a Gatekeeper), a protocol version (IPv4 or IPv6), a test features description (NC for a normal call, FC for a call using a FastConnect procedure, T for a call by using a tunneling procedure, and PC for a call by using a parallel procedure) and a short scenario description, for example, VH_001_B2B_H323v4_NC_Basic_Call.

Most of the provided sample test configurations share the following settings:

- The used H.225 and H245 protocol versions are 0.0.8.2250.0.5 and 0.0.8.245.0.9 respectively.
- The **Graceful ramp down** option is selected.
- The same terminal type (**Terminal entity without MC**) and bandwidth (**64kbps**) settings are used.
- The used Q.931 settings are 1[00000-] and Caller[00000-] for the call originator and 2[00000-] and Called[00000-] for the call terminator activities.
- The **Send Alert** and **Send Call Processing** options are selected for call terminating activities.
- The default codecs and terminal capabilities settings comprise the G.711 ulaw and G.711 alaw codecs.
- The RTP port is defined based on a sequence generator expression for all H323 activities linked to scenario channels that use RTP functions.

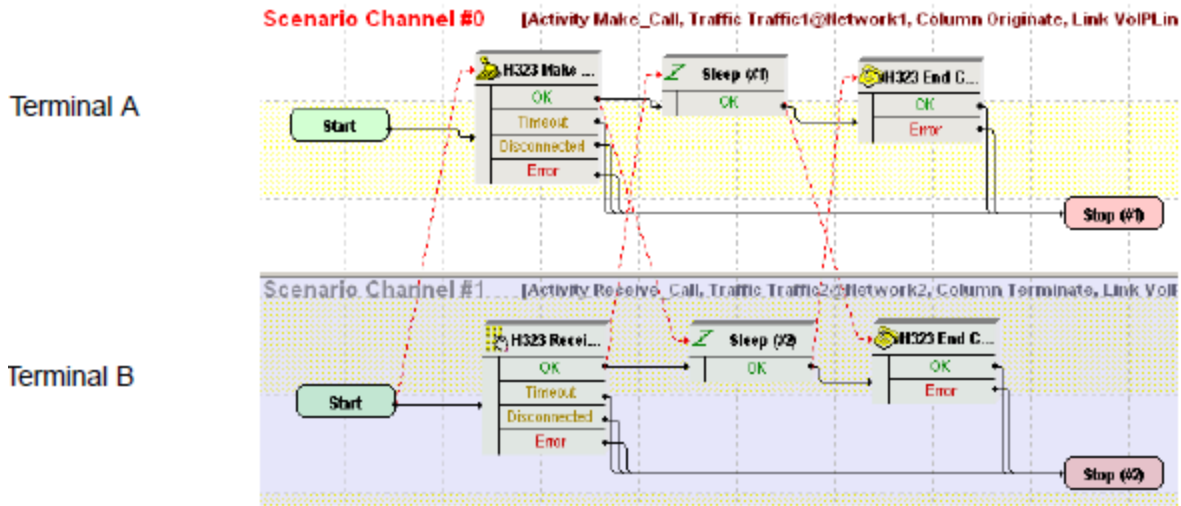
VoIP H.323 Test Configurations

The following sample H.323 test configuration files are contained in the IxLoad installation kit:

VH_001_B2B_H323v4_NC_Basic_Call

This test, which runs in B2B mode, comprises two VoIP H323 activities, Make_Call and Receive_Call, which simulate a number of N phones on each side (N = 100, the test objective value) performing an H323 call without media exchange.

Make_Call is linked to a test scenario channel that originates the call, remains idle for a duration of time specified by the Sleep script function, and then disconnects, as shown in the following image. Receive_Call executes the corresponding call receiving functions flow.



Make_Call configured settings are described in the following table:

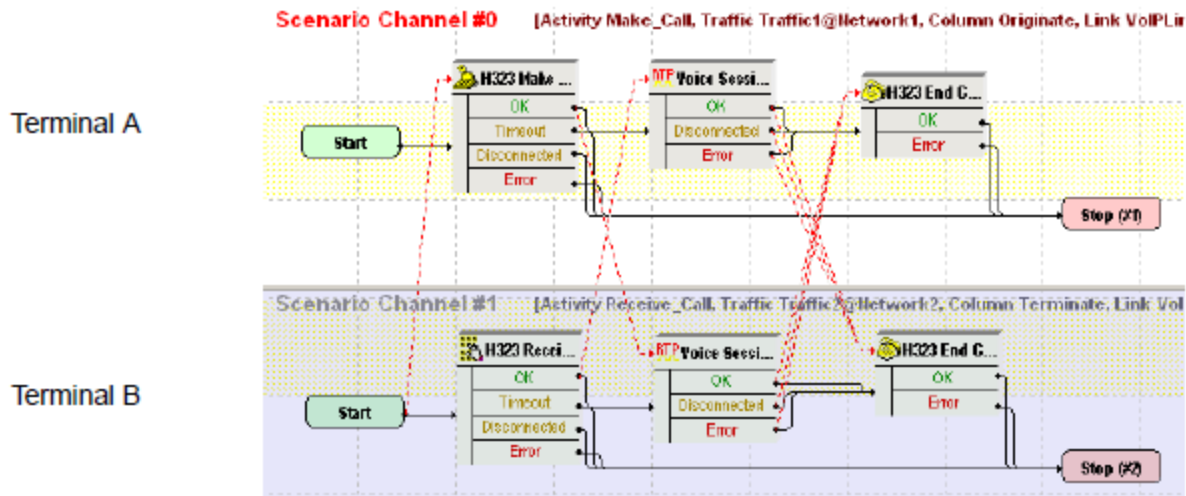
Category	Settings
Scenario	On the call originating scenario channel, H323 terminal A originates a call to H323 terminal B. After call establishment, the call is maintained active for a duration configured by the Sleep function. Eventually, it is terminated by using an H323 End Call function. On the call terminating scenario channel, terminal B executes an H323 Receive Call procedure to answer the call and finally, the H323EndCall procedure for the call terminating side.
Execution	The corresponding scenario channel is configured to execute repeatedly during the test sustain time. The simulated H323 terminals use consecutive phone numbers (per activity).
Dial Plan	The originating H323 phone numbers are defined by using a 160[00000000-] sequence generator expression. The Receive_Call activity is configured as call destination.
H323	The Enable signaling on this activity option is selected for the H323 functions to be executed.
Terminal Capability	The default capabilities settings are used.
Codecs	The default codec settings are used.
RTP	Because this channel does not use any RTP functions, the Enable media on this activity option is not selected.
Other	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the H323 channels.

Note: The Receive_Call activity is configured in a similar way, except that it does not need to specify a call destination.

VH_002_B2B_H323v4_NC_Basic_Call_with_RTP

This test, which runs in B2B mode, comprises two VoIPH323 activities, Make_Call and Receive_Call, which perform an H323 call with bidirectional media exchange.


Make_Call is linked to a test scenario channel that originates the call, performs bidirectional media exchange, and then disconnects, as shown in the following image. Receive_Call executes the call originating functions flow.



Make_Call configured settings are described in the following table:

Category	Settings
Scenario	On the call originating scenario channel, H323 terminal A originates a call to H323 terminal B. After call establishment, media is exchanged bidirectionally by using the VoiceSession script function. On the call terminating scenario channel, terminal B executes a H323 Receive Call procedure to answer the call.
Execution	The corresponding scenario channel is configured to execute repeatedly during the test sustain time. The simulated H323 terminals use consecutive phone numbers (per activity).
Dial Plan	The originating H323 phone numbers are defined by using a 160[00000000-] sequence generator expression. The Receive_Call activity is configured as call destination.
H323	The Enable signaling on this activity option is selected for the H323 functions to be executed.
Terminal Capability	The default capabilities settings are used.

Codecs	The default codec settings are used.
RTP	The Enable media on this activity option is selected.
Other	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the H323 channels.

 **Note:** The Receive_Call activity is configured in a similar way, except that it does not need to specify a call destination.

VH_003_B2B_H323v4_FC_Basic_Call_with_RTP

This test is similar to the previous [VH_002_B2B_H323v4_NC_Basic_Call_with_RTP](#) test, with the difference that the call is established by using the FastStart and Tunneling procedures configured in the H323 page.

VH_004_B2B_H323v4_PC_Basic_Call_with_RTP

This test is similar to the previous [VH_002_B2B_H323v4_NC_Basic_Call_with_RTP](#) test, with the difference that the call uses the FastStart, Tunneling, and Parallel procedures configured in the H323 page.

VH_005_B2B_H323v4_T_Basic_Call_with_RTP

This test is similar to the previous [VH_002_B2B_H323v4_NC_Basic_Call_with_RTP](#) test, with the difference that the call uses the Tunneling procedure configured in the H323 page.

VH_006_B2B_H323v4_NC_Make_Call_with_RTP

This test comprises a single VoIPH323 activity, Make_Call, that simulates a number of H323 phones (N = 100, the test objective value) originating an H323 call to another H323 device. After the call is established, media is exchanged bidirectionally.

Make_Call configured settings are described in the following table:

Category	Settings
Scenario	H323 terminal A originates a call to another H323 terminal. After call establishment, media is exchanged by using the Voice Session script function.
Execution	The corresponding scenario channel is configured to execute repeatedly during the test sustain time. The emulated H323 terminals use consecutive phone numbers (per activity).
Dial Plan	The originating H323 phone numbers are defined by using a 160[00000000-] sequence generator expression. The call destination is specified by using an IP address.
H323	The Enable signaling on this activity option is selected for the H323 functions to be executed.

Terminal Capability	The default capabilities settings are used.
Codecs	The default codec settings are used.
RTP	The Enable media on this activity option is selected.
Other	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the simulated H323 channel.

VH_007_B2B_H323v4_FC_Make_Call_with_RTP

This test is similar to the previous [VH_006_B2B_H323v4_NC_Make_Call_with_RTP](#) test, with the difference that the call uses the FastStart and Tunneling procedures configured in the H323 page.

VH_008_B2B_H323v4_PC_Make_Call_with_RTP

This test is similar to the previous [VH_006_B2B_H323v4_NC_Make_Call_with_RTP](#) test, with the difference that the call uses the **Enable FastStart**, **Enable Tunneling**, and **Enable Parallel H245** options configured in the H323 page.

VH_009_B2B_H323v4_NC_Receive_Call_with_RTP

This test comprises a single VoIPH323 activity, `Receive_Call`, which terminates an H323 call originating from an H323 device. After the call is established, media is exchanged bidirectionally between the two terminals.

`Receive_Call` configured settings are described in the following table:

Category	Settings
Scenario	The simulated H323 terminal terminates an incoming call from another H323 terminal by using an H323 Receive Call function. After call establishment, media is exchanged by using the Voice Session script function. Eventually, it waits for the other party to disconnect and tears down the call.
Execution	The corresponding scenario channel is configured to execute repeatedly during the test sustain time. The emulated H323 terminals use consecutive phone numbers.
Dial Plan	The H323 phone numbers are defined by using a 160[00000000-] sequence generator expression. No call destination is specified.
H323	The Enable signaling on this activity option is selected for the H323 functions to be executed. The Send Call Alerting and Send Call Proceeding options are selected.
Terminal Capability	The default capabilities settings are used.
Codecs	The default codec settings are used.

RTP	The Enable media on this activity option is selected.
Other	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the simulated H323 channel.

VH_010_B2B_H323v4_FC_Receive_Call_with_RTP

This test is similar to the previous [VH_009_B2B_H323v4_NC_Receive_Call_with_RTP](#) test, with the difference that the call uses the FastStart and Tunneling procedures configured in the H323 page.

VH_011_B2B_H323v4_PC_Receive_Call_with_RTP

This test is similar to the previous [VH_009_B2B_H323v4_NC_Receive_Call_with_RTP](#) test, with the difference that the call uses the **Enable FastStart**, **Enable Tunneling**, and **Enable Parallel H245** options configured in the H323 page.

VH_012_B2B_H323v4_NC_Basic_Call_with_RTP_Trunk

This test is similar to the previous [VH_006_B2B_H323v4_NC_Make_Call_with_RTP](#) test, with the following difference at activity configuration level:

- All the simulated originating H323 endpoints (Make_Call activity) use a single IP address.
- The call destination is configured to a single IP address specified in the **Dial Plan** page of the Make_Call activity.
- All terminating H323 endpoints (Receive_Call activity) use another unique IP address, the same as that configured as call destination by the call-originating H323 endpoints.

VH_013_B2B_H323v4_FC_Basic_Call_with_RTP_Trunk

This test is similar to the previous [VH_012_B2B_H323v4_NC_Basic_Call_with_RTP_Trunk](#) test, with the difference that the call uses the FastConnect procedure configured in the H323 page.

VH_014_B2B_H323v4_NC_Basic_Call_with_RTP_all_codecs

This test is similar to the previous [VH_006_B2B_H323v4_NC_Make_Call_with_RTP](#) test, with the difference that all supported codecs are configured in the H323 page and a custom capability descriptor is selected in the **Terminal Capabilities** page.

VH_015_B2B_H323v4_NC_Basic_Call_with_RTP_QoV

This test is similar to the previous [VH_006_B2B_H323v4_NC_Make_Call_with_RTP](#) test, with the difference that the test computes PESQ and P56 QoV scores for a number of 100 channels. The test has the **Enable Qov** option configured in the **RTP** page and the FemaleMale_Mix1 clip on the VoiceSession script functions.

VH_016_B2B_H323v4_NC_Basic_Call_with_HwRTP

This test is similar to the previous [VH_006_B2B_H323v4_NC_Make_Call_with_RTP](#) test, with the difference that the **Enable Hw Acceleration** option is selected in the H323 page of both originating and terminating activities.

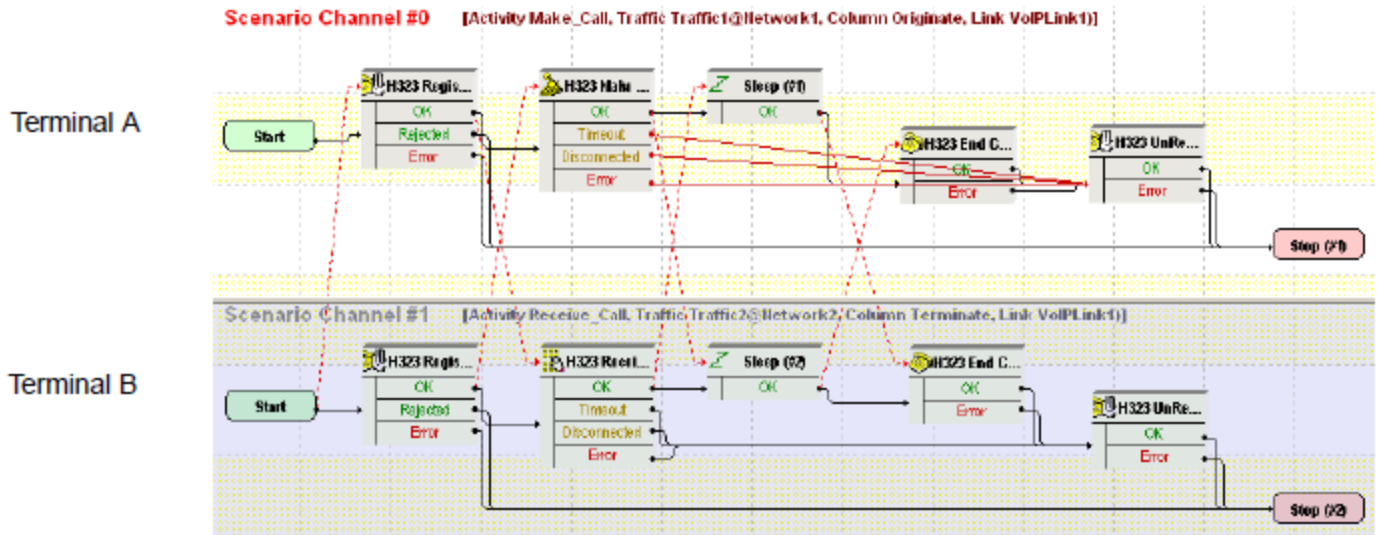
VH_017_B2B_H323v4_NC_Basic_Call_with_HwRTP_10GA_Trunk

This test is similar to the previous [VH_006_B2B_H323v4_NC_Make_Call_with_RTP](#) test, with the difference that the **Allow multiple aggregated 1G ports** option is selected in the **Test Options** configuration page.

VH_018_GK_H323v4_NC_Basic_Call

This test, which runs against a GK, comprises two VoIP H323 activities, *Make_Call* and *Receive_Call*, which simulate a number of N phones (N = 100, the test objective value) performing an H323 call without media exchange.


Make_Call is linked to a test scenario channel that originates the call, remains idle for a user-configured duration of time, and then disconnects, as shown in the following image. *Receive_Call* executes the call originating functions flow.



Make_Call configured settings are described in the following table:

Category	Settings
Scenario	On the call originating scenario channel, H323 terminal A registers with a GK and originates a call to H323 terminal B. After call establishment, the terminal stays idle for a configured period of time, and then disconnects the call. Eventually, the terminal unregisters with the GK. On the call terminating scenario channel, terminal B first registers with the GK, and then executes a H323 Receive Call procedure to answer the call.
Execution	The corresponding scenario channel is configured to execute repeatedly during the test sustain time. The simulated H323 terminals use consecutive phone numbers (per activity).
Dial Plan	The originating H323 phone numbers are defined by using a 160[00000000-] sequence generator expression. The <i>Receive_Call</i> activity is configured as call destination.

H323	The Enable signaling on this activity option is selected for the H323 functions to be executed. The Enable RAS, Use Registration parameters, Use Gatekeeper for admission, Enable Disengage, Enable Keep-Alive registration options are selected. The IP address of the GK test is running against needs specified in the Gatekeeper area.
Terminal Capability	The default capabilities settings are used.
Codecs	The default codec settings are used.
RTP	The Enable media on this activity option is not selected.
Other	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the H323 channels.

 **Note:** The Receive_Call activity is configured in a similar way, except that it does not need to specify a call destination.

VH_019_GK_H323v4_NC_Basic_Call_with_RTP

This test is similar to the previous [VH_018_GK_H323v4_NC_Basic_Call](#) test, with the difference that after call establishment, bidirectional media transfer is performed by using the VoiceSession script function. As such, the **Enable media on this activity** option is selected for both the Make_Call and the Receive_Call activities.

VH_020_GK_H323v4_FC_Basic_Call_with_RTP

This test is similar to the previous [VH_019_GK_H323v4_NC_Basic_Call_with_RTP](#) test, with the difference that the call uses the **Enable FastStart** and **Enable Tunneling** options configured in the H323 page. After call establishment, bidirectional media transfer is performed by using the VoiceSession script function.

VH_021_GK_H323v4_PC_Basic_Call_with_RTP

This test is similar to the previous [VH_019_GK_H323v4_NC_Basic_Call_with_RTP](#) test, with the difference that the call uses the **Enable FastStart, Enable Tunneling, and Enable Parallel H245** options configured in the H323 page. After call establishment, bidirectional media transfer is performed by using the VoiceSession script function.

VH_022_GK_H323v4_T_Basic_Call_with_RTP


This test is similar to the previous [VH_019_GK_H323v4_NC_Basic_Call_with_RTP](#) test, with the difference that the call uses the **Enable Tunneling** option configured in the H323 page. After call establishment, bidirectional media transfer is performed by using the VoiceSession script function.


VH_023_GK_H323v4_NC_Basic_Call_with_RTP_Trunk

This test comprises two VoIP H323 activities, Make_Call and Receive_Call, which simulate a number of N (N = 100, the test objective value) H323 terminals performing an H323 call with media exchange. The test runs against a user-specified Gatekeeper as the DUT.

The Make_Call configured settings are described in the following table:

Category	Settings
Scenario	H323 terminal A registers with a GK, and then originates a call to the H323 terminal B. After call establishment, media is exchanged bidirectionally by using the Voice Session script function. Eventually, terminal A tears down the call.
Execution	The corresponding scenario channel is configured to execute continuously during the test sustain time. The emulated H323 terminals use consecutive phone numbers (per activity).
Dial Plan	The H323 phone numbers are defined by using a 160[00000000-] sequence generator expression. Receive_Call is configured as call destination.
H323	The Enable signaling on this activity option is selected for the H323 functions to be executed. The GK IP address is configured in the Gatekeeper area.
Terminal Capability	The default capabilities settings are used.
Codecs	The default codec settings are used.
RTP	The Enable media on this activity option is selected. Because all RTP endpoints use a single IP address, a [10000-65535, 2] sequence generating expression needs to be specified in the RTP Port field.
Other	The IP version preference is set to IPv4, and no scenario variables need to be initialized for the H323 channels.

 **Note:** The Receive_Call activity is configured in a similar way, except that it does not need to specify a call destination.

 **Note:** This test is different from other sample tests in that the Make_Call simulated H323 endpoints use a single IP address. This also holds true for the Make_Call simulated H323 and RTP endpoints.

VH_024_GK_H323v4_FC_Basic_Call_with_RTP_Trunk

This test is similar to the previous [VH_019_GK_H323v4_NC_Basic_Call_with_RTP](#) test, with the difference that the call uses the **FastStart** and **Enable Tunneling** options configured in the H323 page.

VH_025_GK_H323v4_NC_Basic_Call_with_RTP_all_codecs

This test is similar to the previous [VH_019_GK_H323v4_NC_Basic_Call_with_RTP](#) test, with the difference that all supported codecs are configured in the H323 page and a custom capability descriptor is selected in the **Terminal Capabilities** page.

VH_026_GK_H323v4_NC_Basic_Call_with_RTP_QoS


This test is similar to the previous [VH_019_GK_H323v4_NC_Basic_Call_with_RTP](#) test, with the difference that the test computes PESQ and P56 QoV metrics. The test has the **Enable QoV** option configured in the **RTP** page of both originating and terminating activities, and the FemaleMale_Mix1 clip on the VoiceSession script functions.

VH_027_GK_H323v4_NC_Basic_Call_with_HwRTP

This test is similar to the previous [VH_019_GK_H323v4_NC_Basic_Call_with_RTP](#) test, with the difference that the **Enable Hw Acceleration** option is selected in the H323 page of both originating and terminating activities.

VH_028_GK_H323v4_NC_Basic_Call_with_HwRTP_10GA_ER_Trunk

This test is similar to the previous [VH_027_GK_H323v4_NC_Basic_Call_with_HwRTP](#) test, with the difference that the **Allow multiple aggregated 1G ports** option is selected in the **Test Options** configuration page.

 **Note:** The rest of the H323 samples, ranging from VH_029 to VH_039, represent implementations of some of the previous tests by using IPv6 instead of IPv4 network level settings.

H.248 Sample Test Configuratings and Test Scenarios

This section describes the pre-defined IxLoad Voice Plug-in H.248/MEGACO available sample test configurations (RXFs) and their associated test scenarios.

Note: For a complete description of the supported H.248 Test Library functions, see [VoIP H248 Functions Library](#).

Note: Sample tests follow a naming convention that comprises test type (VM for VoIP MEGACO), an index, a test topology (T1 or T2 shown in the following two images respectively), an IP protocol version (IPv4 or IPv6), a configuration (B2B or vs_DUT), and a short call description, for example, VM_001_H248_IPv4_B2B_with_version_3.

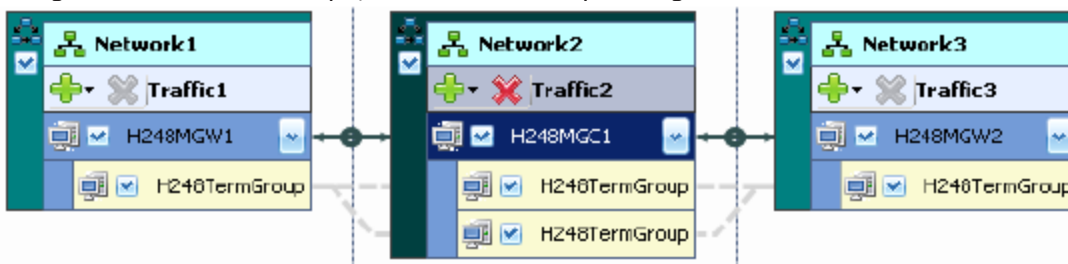
Used Test Configurations

Most of the sample H248 tests supplied with IxLoad are based on one of the following two configurations:

- **MGW/MGC:** IxLoad simulates a single MGW and its controlling MGC. The simulated H248 protocol and RTP functions flow is mapped to H248TermGroups that are associated with each of the H248MGW1 and H248MGC1 activities.



- **Two MGWs/MGC:** IxLoad simulates two MGW and their controlling MGC. The simulated H248 protocol and RTP functions flow is mapped to H248TermGroups that are associated with each of the H248MGW1, H248MGW1, and H248MGC1 activities. Note that the H248MGC1 activity has two configured H248TermGroups, each one corresponding to a controlled GW.



VoIP H.248 Test Configurations

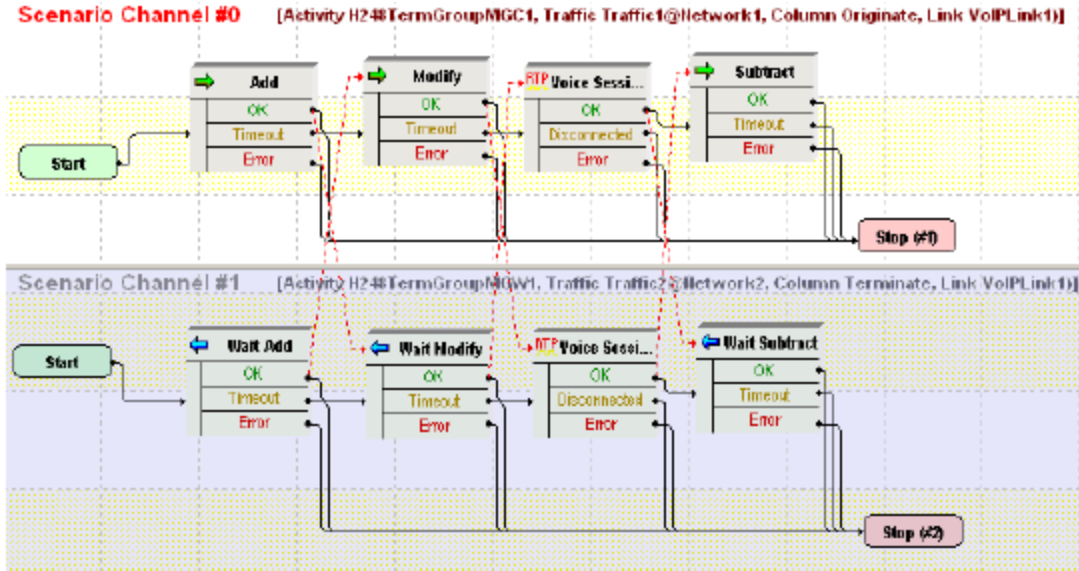
The following sample VoIP H.248 test configuration files are contained in the IxLoad installation kit:

VM_001_H248_IPv4_B2B_with_version_v3

This test based on the configuration shown in T1 simulates an Access Gateway, with a configured objective of two channels, and a controlling MGC. The GW activity has a TermGroup defined that

executes an H.248 signaling and media functions flow. The MGC has an associated TermGroup that executes an H248 and media functions flow.

The underlying two-channel test scenario involving H248TermGroupMGC1 and H248TermGroupMGW1 is shown in the following image:



The H248MGC1 and H248TermGroupMGC1 configured settings are described in the following table:

Category	Settings
H248MGC1	
Simulated MGC	The controlled GW type is configured to Access Gateway (PSTN2IP) and the H248TermGroupMGC1 is enabled.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to ServiceChange requests, Auto-reply to Notify requests, Send Modify on Root termination to set properties, Use TransactionResponseAck , and Wait for MGW registration options are selected.
Profiles	The ETSI_ARGW/1 profile is selected.
H248TermGroupMGC1	
Scenario	The test scenario comprises the Add and Modify functions followed by the VoiceSession function for media exchange. The final Subtract function performs the deletion of all active terminations.

Execution Settings	The corresponding scenario channel is configured to execute twice during the test sustain time.
SDP	No overriding SDP settings are configured.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

The H248MGW1 and H248TermGroupMGW1 configured settings are described in the following table:

Category	Settings
H248MGW1	
Simulated MGW	The simulated GW type is configured to Access Gateway and the controlling MGC is set to the H248MGC1 activity.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to ServiceChange requests, Auto-reply to Audit requests, Auto-reply to Modify on Root termination, Use TransactionResponseAck, and the Auto-register options are selected.
Profiles	The ETSI_ARGW/1 profile is selected for the simulated MGW.
H248TermGroupMGW1	
Scenario	The test scenario comprises the Wait Add and Wait Modify messages followed by a media session by using the VoiceSession function. The last function is a Wait Subtract function.
Execution Settings	The corresponding scenario channel is configured to execute twice during the test sustain time.
SDP	No overriding SDP settings are configured.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

VM_002_H248_IPv4_B2B_with_version_v2

This test is similar with that described in [VM_001_H248_IPv4_B2B_with_version_v3](#), except that it uses version 2 of the H.248/MEGACO protocol.

VM_003_H248_IPv4_B2B_with_version_v1

This test is similar with that described in [VM_001_H248_IPv4_B2B_with_version_v3](#), except that it uses version 1 of the H.248/MEGACO protocol.

VM_004_H248_IPv6_B2B_with_version_v3

This test is similar with that described in [VM_001_H248_IPv4_B2B_with_version_v3](#), except that it uses IPv6 addressing instead of IPv4.

VM_005_H248_IPv6_B2B_with version_v2

This test is similar with that described in [VM_002_H248_IPv4_B2B_with_version_v2](#), except that it uses IPv6 addressing instead of IPv4.

VM_006_H248_IPv6_B2B_with_version_v1

This test is similar with that described in [VM_003_H248_IPv4_B2B_with_version_v1](#), except that it uses IPv6 addressing instead of IPv4.

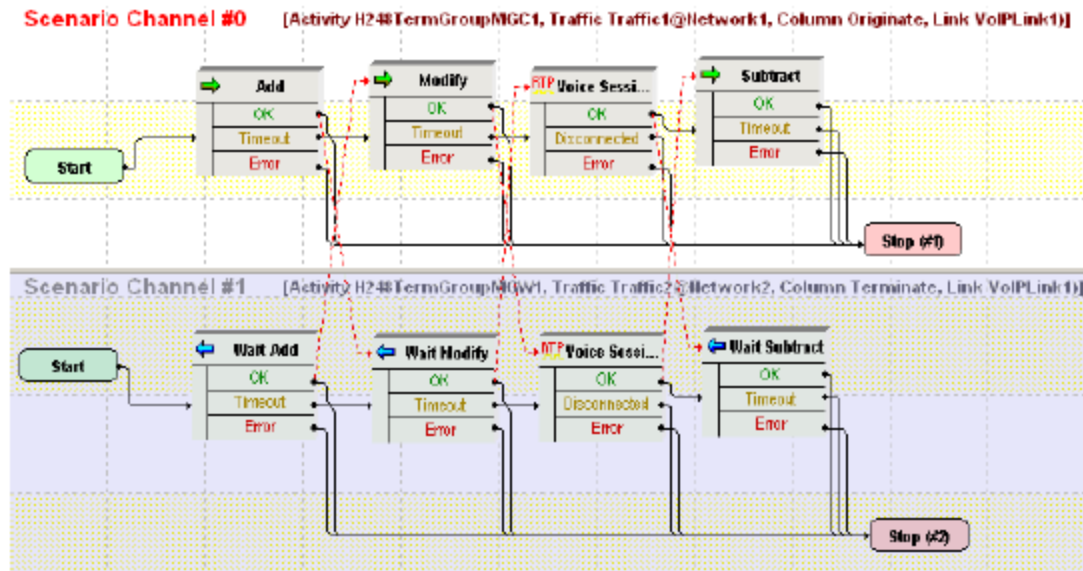
VM_007_H248_IPv4_B2B_with_message_maximum_size_4000

This test is similar with that described in [VM_001_H248_IPv4_B2B_with_version_v3](#), except that it uses a maximum H.248 message size of 4000 bytes.

VM_008_H248_IPv4_B2B_reply_send_AuditValue

This test based on the configuration shown in T1 simulates a Trunking GW with a configured objective of 100 channels and a controlling MGC. The MGW activity has a TermGroup defined that executes an H.248 signaling and media functions flow. The MGC has an associated TermGroup that executes an H248 and media functions flow.

The underlying two-channel test scenario involving H248TermGroupMGC1 and H248TermGroupMGW1 is shown in the following image:



The H248MGC1 and H248TermGroupMGC1 configured settings are described in the following table:

Category	Settings
H248MGC1	
Simulated MGC	The controlled GW type is configured to Trunking Gateway (PSTN2IP) and the H248TermGroupMGC1 is enabled.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to ServiceChange requests , Auto-reply to Notify requests , Send Modify on Root termination to set properties , and Use TransactionResponseAck options are selected. The Wait for MGW Registration and the Send AuditValue options are also selected.
Profiles	The ETSI_ARGW/1 profile is selected.
H248TermGroupMGC1	
Scenario	The test scenario comprises the Add and Modify (auto SDP) functions followed by the VoiceSession function for media exchange. Eventually, a Subtract function performs the deletion of all active terminations.
Execution Settings	The corresponding scenario channel is configured to execute five times during the test sustain time.
SDP	No overriding SDP settings are configured.

Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

The H248MGW1 and H248TermGroupMGW1 configured settings are described in the following table:

Category	Settings
H248MGW1	
Simulated MGW	The simulated GW type is configured to Trunking Gateway and the controlling MGC is set to the H248MGC1 activity.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to ServiceChange requests, Auto-reply to Audit requests, Auto-reply to Modify on Root termination, Use TransactionResponseAck, and the Auto-register options are selected.
Profiles	The ETSI_TGW/1 profile is selected for the simulated MGW.
H248TermGroupMGW1	
Scenario	The test scenario comprises the Wait Add and Wait Modify messages followed by a media session by using the VoiceSession function. The last function is a Wait Subtract function.
Execution Settings	The corresponding scenario channel is configured to execute five times during the test sustain time.
SDP	No overriding SDP settings are configured.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

VM_009_H248_IPv4_B2B_reply_send_AuditCapabilities

This test is similar with that described in [VM_008_H248_IPv4_B2B_reply_send_AuditValue](#), except that it has the **Send AuditCapabilities** instead of the **Send AuditValue** option is configured in the Automatic Functionality page of the H248MGC1 activity.

VM_010_H248_IPv4_B2B_enable_retransmissions

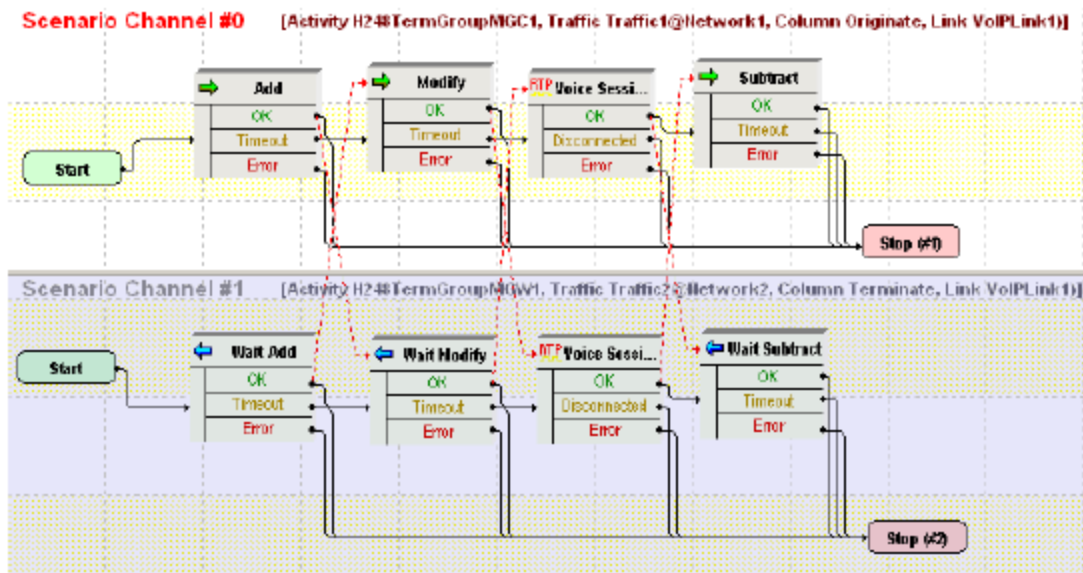
This test is similar with that described in [VM_009_H248_IPv4_B2B_reply_send_AuditCapabilities](#), except that it has the **Enable Retransmissions** option configured on the H248MGC1 activity (Automatic Functionality page).

At test scenario level, the Wait Add script function on the H248MGW1 activity has a configured delay of 600 ms, such as to enforce the retransmission of MGC Add messages.

VM_011_H248_IPv4_B2B_Access_gw

This test based on the configuration shown in T1 simulates an Access GW with a configured objective of 100 channels and a controlling MGC. The GW activity has a TermGroup defined that executes an H.248 signaling and media functions flow. The MGC has an associated TermGroup that executes an H248 and media functions flow.

The underlying two-channel test scenario involving H248TermGroupMGC1 and H248TermGroupMGW1 is shown in the following image:



The H248MGC1 and H248TermGroupMGC1 configured settings are described in the following table:

Category	Settings
H248MGC1	
Simulated MGC	The controlled GW type is configured to Trunking Gateway (PSTN2IP) and the H248TermGroupMGC1 is enabled.
H248	Version 3 of the H.248/MEGACO protocol is configured.

Automatic Functionality	The Auto-reply to ServiceChange requests, Auto-reply to Notify requests, Send Modify on Root termination to set properties, and Use TransactionResponseAck, and Wait for MGW Registration options are selected.
Profiles	The ETSI_ARGW/1 profile is selected.
H248TermGroupMGC1	
Scenario Editor	The test scenario comprises the Add and Modify (auto SDP descriptor) functions followed by the VoiceSession function for media exchange. The final Subtract function performs the deletion of all active terminations.
Execution Settings	The corresponding scenario channel is configured to execute five times during the test sustain time.
SDP	The auto SDP option at script function level is not overridden by any setting in this tab.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

The H248MGW1 and H248TermGroupMGW1 configured settings are described in the following table:

Category	Settings
H248MGW1	
Simulated MGW	The simulated GW type is configured to Access Gateway and the controlling MGC is set to the H248MGC1 activity. H248TermGroupMGW1 is configured on the H248MGW1 activity.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to ServiceChange requests, Auto-reply to Audit requests, Auto-reply to Modify on Root termination, Use TransactionResponseAck, and the Auto-register options are selected.
Profiles	The ETSI_TGW/1 profile is selected for the simulated MGW.
H248TermGroupMGW1	

Scenario	The test scenario comprises the Wait Add and Wait Modify messages followed by a media session by using the VoiceSession function. The last function is a Wait Subtract function.
Execution Settings	The corresponding scenario channel is configured to execute five times during the test sustain time.
SDP	No overriding SDP settings are configured.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

VM_012_H248_IPv4_B2B_Border_gw

This test based on the configuration shown in T1 simulates a Border GW (IP2IP) with a configured objective of 10 channels, a controlling MGC, and four media-only VoIPSIPPeer activities that simulate RTP endpoints.

The MGC has an associated H248TermGroup that executes an H248-only functions flow. An RTP endpoints pair defined by using two media-only VoIPSIPPeer activities simulate the RTP terminations associated with the MGC.

The MGW activity has an associated H248TermGroup that executes an H.248-only functions flow. An RTP endpoints pair defined by using another two media-only VoIPSIPPeer activities simulate the RTP terminations associated with the MGW.

The media-only VoIPSIPPeer activities VoIPSIPPeer1 to VoIPSIPPeer4 are defined on four NetTraffics, each with a different underlying network range, and simulate RTP endpoints. Each RTP endpoints pair, VoIPSIPPeer2 – VoIPSIPPeer3 and VoIPSIPPeer1 – VoIPSIPPeer4 exchange media bidirectionally.

The test configuration has two associated test scenarios:

- The first scenario VM_012_H248_IPv4_B2B_Border_gw_MGC comprises five channels:
 - Channel#0: Executes an MGC H248 functions flow.
 - Channel#1 to channel#4: Each channel executes a bidirectional media exchange by using a VoiceSession script function.
- The second scenario VM_012_H248_IPv4_B2B_Border_gw_MGW comprises a single channel that executes an H.248 functions flow mirroring that from channel#0 of the VM_012_H248_IPv4_B2B_Border_gw_MGC test scenario.

The H248MGC1 and H248TermGroupMGC1 configured settings are described in the following table:

Category	Settings
H248MGC1	

Simulated MGC	The controlled GW type is configured to Border Gateway (IP2IP) and H248TermGroupMGC1 is enabled.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Wait for Registration option is selected.
Profiles	The ETSI_ARGW/1 profile is selected.
H248TermGroupMGC1	
Scenario	The activity is linked to the first channel of the VM_012_H248_IPv4_B2B_Border_gw_MGC test scenario comprising the following script functions: <ul style="list-style-type: none"> • Sleep • Add • Modify: Conveys to the first RTP termination a remoteDescriptor with a custom SDP definition based on the VOIP_Var0 and VOIP_IPAddress0 variables. Conveys to the second RTP termination a remoteDescriptor with a custom SDP definition based on the VOIP_Var1 and VOIP_IPAddress1 variables. • Subtract • Sleep
Execution Settings	The corresponding scenario channel is configured to execute three times during the test sustain time.
SDP	No overriding SDP settings are used. The Skip SDP processing option is not selected.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is not selected.
Other Settings	The scenario variables are used as follows: <ul style="list-style-type: none"> • VOIP_IPAddress0 and VOIP_Var0 are initialized to the IP address and port values of the VoIPSIPPeer3-emulated RTP endpoints. • VOIP_IPAddress1 and VOIP_Var1 are initialized to the IP address and port values of the VoIPSIPPeer1-emulated RTP endpoints. • VOIP_IPAddress2 and VOIP_Var2 are initialized to the IP address and port values of the VoIPSIPPeer2-emulated RTP endpoints. • VOIP_IPAddress2 and VOIP_Var2 are initialized to the IP address and port values of the VoIPSIPPeer4-emulated RTP endpoints.

The H248MGW1 and H248TermGroupMGW1 configured settings are described in the following table:

Category	Settings
H248MGW1	
Simulated MGW	The controlled GW type is configured to Border Gateway (IP2IP) and H248TermGroupMGC1 is enabled.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto Register option is selected.
Profiles	The ETSI_GateControl/1 profile is selected.
H248TermGroupMGW1	
Scenario	<p>The activity is linked to the first channel of the VM_012_H248_IPv4_B2B_Border_gw_MGW test scenario comprising the following script functions:</p> <ul style="list-style-type: none"> • Sleep • Wait Add: For the first RTP termination, it returns a localDescriptor with a custom SDP definition, based on the VOIPVar2 and VOIP_IPAddress2 variables configured in the H248MGW1 activity. For the second RTP termination, it returns localDescriptor with a custom SDP definition, based on the VOIPVar3 and VOIP_IPAddress3 variables configured in the H248MGW1 activity. • Modify • Wait Subtract • Sleep
Execution Settings	The corresponding scenario channel is configured to execute three times during the test sustain time.
SDP	No overriding SDP settings are used.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is not selected.
Other Settings	<p>The scenario variables are used as follows:</p> <ul style="list-style-type: none"> • VOIP_IPAddress0 and VOIP_Var0 are initialized to the IP address and port values of the VoIPSIPPeer3-emulated RTP endpoints. • VOIP_IPAddress1 and VOIP_Var1 are initialized to the IP address and port values of the VoIPSIPPeer1-emulated RTP endpoints. • VOIP_IPAddress2 and VOIP_Var2 are initialized to the IP address and port values of the VoIPSIPPeer2-emulated RTP endpoints. • VOIP_IPAddress2 and VOIP_Var2 are initialized to the IP address and port values of the VoIPSIPPeer4-emulated RTP endpoints.

The configuration of all four simulated VoIPSIPPeer activities is similar to that of the VoIPSIPPeer1 activity, described in the following table:

Category	Settings
Scenario	The underlying channel comprises a single VoiceSession function for bidirectional media exchange.
Execution Settings	The following RTP Channel Mapping settings are used: <ul style="list-style-type: none"> • IP: Use Consecutive values (per port) • UDP port: Use consecutive values (per port)
Dial Plan	The VoIPSIPPeer4 : <portrange> activity is configured as call destination. The portrange value is that configured for the VOIP_Var3 variable in the Other Settings tab of the H248MGC1 activity.
SIP Settings	The use of signaling is not selected.
Codec Settings	The default codec settings are used.
RTP Settings	The Enable media on this activity option is selected and the RTP port is defined by using a sequence generator expression.
Other Settings	The IP version preference is set to IPv4, and no scenario variables need to be initialized.

VM_013_H248_IPv4_B2B_Trunking_gw

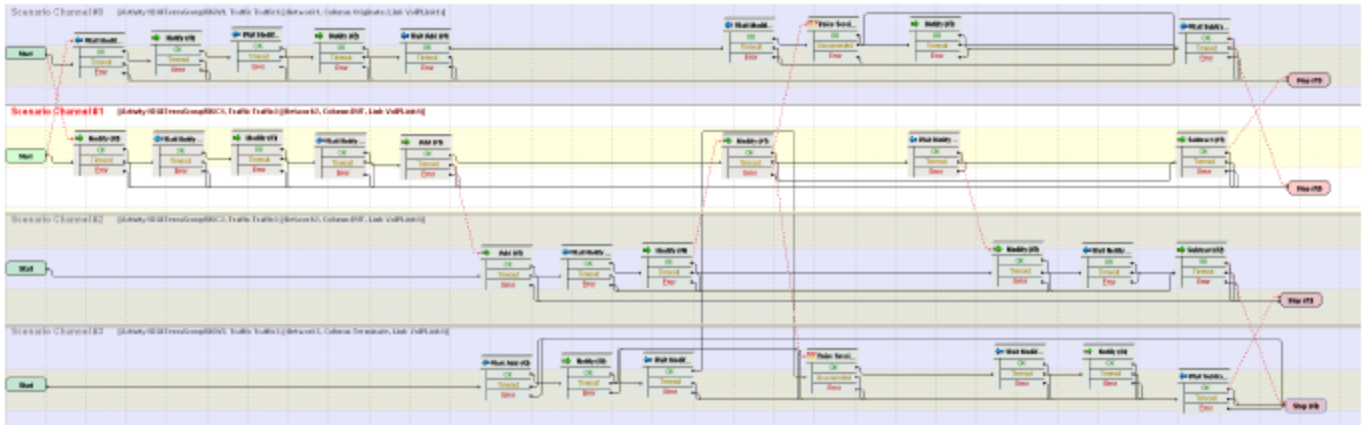
This test is similar with that described in [VM_011_H248_IPv4_B2B_Access_gw](#), except that it simulates a Trunking Gateway instead of an Access Gateway.

VM_014_H248_IPv4_B2B_Residential_gw

This test based on the configuration shown in [Used Test Configurations](#) simulates two Residential GWs, each with a with a configured objective of 10 channels, and a controlling MGC.

Each MGW activity has an H248TermGroup defined that executes an H.248 and media functions flow. The MGC has two associated TermGroups that execute an H248-only functions flow.

The underlying four-channel test scenario involving H248TermGroupMGW1, H248TermGroupMGW2, H248TermGroupMGC1, and H248TermGroupMGC2 is shown in the following image:



The H248MGC1 and H248TermGroupMGC1 configured settings are described in the following table:

Category	Settings
H248MGC1	
Simulated MGC	The controlled GW type is configured to Residential Gateway (PSTN2IP) and the H248TermGroupMGC1 is enabled.
H248	Version 1 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Wait for Registration option is selected.
Profiles	The ETSI_ARGW/1 profile is selected.
H248TermGroupMGC1	
Scenario	The test scenario channel comprises the following functions: <ul style="list-style-type: none"> • Modify • Wait Notify • Modify • Wait Notify • Add: the received localDescriptor value is stored into the RemoteDescriptor variable. • Modify: the sent remoteDescriptor has a custom definition that uses the \$RemoteDescriptor[\$arrScenarioCh[2]] variable corresponding to the SDP value from the scenario channel #2. • Wait Notify • Subtract
Execution Settings	The corresponding scenario channel is configured to execute repeatedly during the test sustain time.

SDP	The Skip SDP processing option is selected.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is not selected.
Other Settings	No scenario variables need to be initialized.

The H248MGW1 and H248TermGroupMGW1 configured settings are described in the following table:

Category	Settings
H248MGW1	
Simulated MGW	The simulated GW type is configured to Residential Gateway and the controlling MGC is set to the H248MGC1 activity. H248TermGroupMGW1 is configured on the MGW activity.
H248	Version 1 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to Modify on Root termination and Auto Register options are selected.
Profiles	The ETSI_TGW/1 profile is selected for the simulated MGW.
H248TermGroupMGW1	
Scenario	The test scenario channel comprises the following functions: <ul style="list-style-type: none"> • Wait Modify • Notify • Wait Modify • Notify • Wait Add • Wait Modify • VoiceSession: Performs bidirectional media exchange • Notify: Notifies the MGC of an onhook condition • Wait Subtract
Execution Settings	The corresponding scenario channel is configured to execute repeatedly during the test sustain time.
SDP	No overriding SDP settings are configured.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.

RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

Note: H248MGW2 and H248TermGroupMGW2 are configured similar to H248MGW1 and H248TermGroupMGW1 respectively, with the exception of the underlying test scenario channel. The Add function stores the received localDescriptor value in the RemoteDescriptor variable and sends a custom remoteDescriptor based on the \$RemoteDescriptor[\$arrScenarioCh[1]] variable corresponding to the SDP value from scenario channel #1. H248TermGroupMGC2 is configured similar to H248TermGroupMGC1.

VM_015_H248_IPv4_B2B_MID_IP_Address

This test based on the configuration shown in T1 simulates a Trunking Gateway, with a configured objective of 10 channels, and a controlling MGC.

Both the H248MGW1 and H248MGC1 activity have H248TermGroups defined that execute an H.248 and RTP functions flow.

The H248MGC1 and H248TermGroupMGC1 configured settings are described in the following table:

Category	Settings
H248MGC1	
Simulated MGC	The MID format parameter is configured to the IP Address setting. The controlled GW type is configured to Trunking Gateway (PSTN2IP) and the H248TermGroupMGC1 is enabled.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to ServiceChange requests, Auto-reply to Notify requests, Send Modify on Root termination to set properties, Use TransactionResponseAck, and Wait for MGW registration options are selected.
Profiles	The ETSI_ARGW/1 profile is selected.
H248TermGroupMGC1	
Scenario	The test scenario channel comprises an Add and a Modify function. Media exchange is performed by using a VoiceSession function. Eventually, the terminations are disconnected by using a Subtract function.
Execution Settings	The corresponding scenario channel is configured to execute three times during the test sustain time.
SDP	No overriding SDP settings are configured.

Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

The H248MGW1 and H248TermGroupMGW1 configured settings are described in the following table:

Category	Settings
H248MGW1	
Simulated MGW	The MID format parameter is configured to the IP address setting. The simulated GW type is configured to Trunking Gateway and the controlling MGC is set to the H248MGC1 activity. H248TermGroupMGW1 is configured on the MGW activity.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to ServiceChange requests, Auto-reply to Audit requests, Auto-reply to Modify on Root termination, Use TransactionResponseAck, and the Auto-register options are selected.
Profiles	The ETSI_TGW/1 profile is selected for the simulated MGW.
H248TermGroupMGW1	
Scenario	The test scenario channel comprises the script functions corresponding to the MGC-transmitted commands .
Execution Settings	The corresponding scenario channel is configured to execute three times during the test sustain time.
SDP	No overriding SDP settings are configured.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

VM_016_H248_IPv4_B2B_MID_Device_Name

This test is similar to [VM_015_H248_IPv4_B2B_MID_IP_Address](#), except that it uses a Device name option for the MID parameter.

VM_017_H248_IPv4_B2B_MID_IPAddress_Port

This test is similar to [VM_015_H248_IPv4_B2B_MID_IP_Address](#), except that it uses an **IP Address: Port** option for the MID parameter.

VM_018_H248_IPv4_B2B_MID_MGW_MGC_DNS_Name

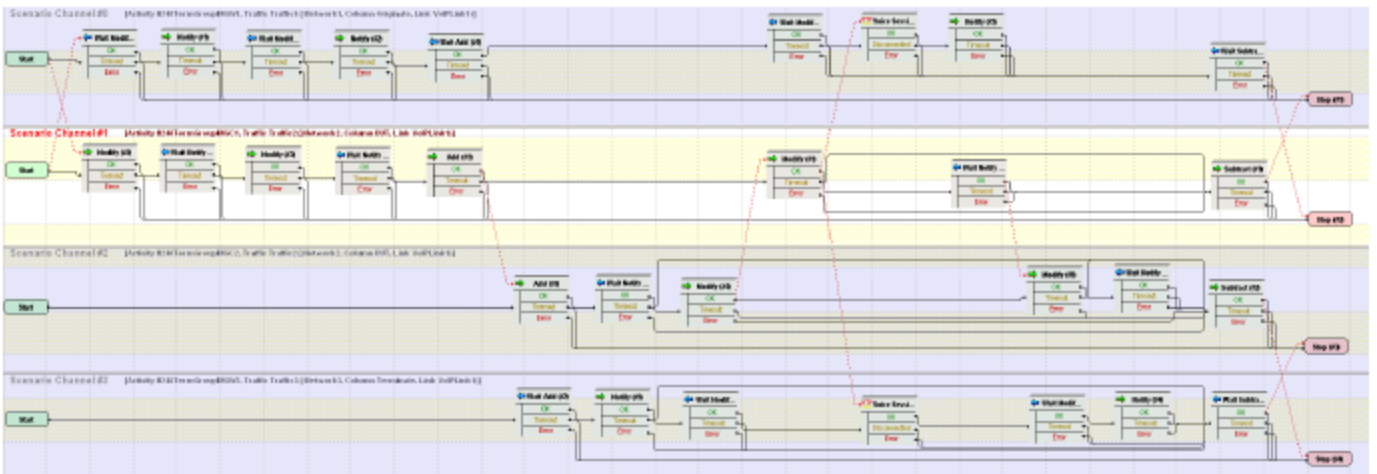
This test is similar to [VM_015_H248_IPv4_B2B_MID_IP_Address](#), except that it uses an MGC DNS name/MGW DNS name option for the MID parameter.

VM_019_H248_IPv4_B2B_performance_topology2_rtp

This test based on the configuration shown in T2 simulates two Residential GWs, each with a with a configured objective of 900 channels, and a controlling MGC.

Each MGW activity has an H248TermGroup defined that executes an H.248 and media functions flow. The MGC has two associated TermGroups that execute an H248-only functions flow.

The underlying four-channel test scenario involving H248TermGroupMGW1, H248TermGroupMGW2, H248TermGroupMGC1, and H248TermGroupMGC2 is shown in the following image:



The H248MGC1 and H248TermGroupMGC1 configured settings are described in the following table:


Category	Settings
H248MGC1	
Simulated MGC	The controlled GW type is configured to Residential Gateway (PSTN2IP) and the H248TermGroupMGC1 is enabled.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Wait for MGW Registration option is selected.
Profiles	The ETSI_TGW/1 profile is selected.

H248TermGroupMGC1	
Scenario	The test scenario channel comprises the following functions: <ul style="list-style-type: none"> • Modify • Wait Notify • Modify • Wait Notify • Add: the received localDescriptor value is stored into the RemoteDescriptor variable. • Modify: the sent remoteDescriptor has a custom definition that uses the \$RemoteDescriptor[\$arrScenarioCh[2]] variable corresponding to the SDP value from the scenario channel #2. • Wait Notify • Subtract
Execution Settings	The corresponding scenario channel is configured to execute three times during the test sustain time.
SDP	The Skip SDP processing option is selected.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is not selected.
Other Settings	No scenario variables need to be initialized.

The H248MGW1 and H248TermGroupMGW1 configured settings are described in the following table:

Category	Settings
H248MGW1	
Simulated MGW	The simulated GW type is configured to Residential Gateway and the controlling MGC is set to the H248MGC1 activity. H248TermGroupMGW1 is configured on the MGW activity.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to Modify on Root termination and Auto Register options are selected.
Profiles	The ETSI_TGW/1 profile is selected for the simulated MGW.
H248TermGroupMGW1	

Scenario	The test scenario channel comprises the following functions: <ul style="list-style-type: none"> • Wait Modify • Notify • Wait Modify • Notify • Wait Add • Wait Modify • VoiceSession: Performs bidirectional media exchange • Notify: Notifies the MGC of an onhook condition • Wait Subtract
Execution Settings	The corresponding scenario channel is configured to execute three times during the test sustain time.
SDP	No overriding SDP settings are configured.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

 **Note:** H248MGW2 and H248TermGroupMGW2 are configured similar to H248MGW1 and H248TermGroupMGW1 respectively, with the exception of the underlying test scenario channel. The Add function stores the received localDescriptor value in the RemoteDescriptor variable and sends a custom remoteDescriptor based on the \$RemoteDescriptor[\$arrScenarioCh[1]] variable corresponding to the SDP value from scenario channel #1. H248TermGroupMGC2 is configured similar to H248TermGroupMGC1.

VM_020_H248_IPv4_B2B_performance_topology1_rtp

This test based on the configuration shown in T1 simulates a Trunking GW, with a configured objective of 900 channels, and a controlling MGC.

Both the H248MGW1 and H248MGC1 activity have H248TermGroups defined that execute an H.248 and media functions flow.

The H248MGC1 and H248TermGroupMGC1 configured settings are described in the following table:

Category	Settings
H248MGC1	
Simulated MGC	The controlled GW type is configured to Trunking Gateway (PSTN2IP) and the H248TermGroupMGC1 is enabled.

H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to ServiceChange requests, Auto-reply to Notify requests, Send Modify on Root termination to set properties, Use TransactionResponseAck , and Wait for MGW registration options are selected.
Profiles	The ETSI_TGW/1 profile is selected.
H248TermGroupMGC1	
Scenario	The test scenario channel comprises two Modify functions for the physical terminations, followed by an Add and a Modify function for the RTP terminations. Media is exchanged bidirectionally by using a VoiceSession function. Eventually, the terminations are disconnected by using a Subtract function.
Execution Settings	The corresponding scenario channel is configured to execute repeatedly during the test sustain time.
SDP	No overriding SDP settings are configured.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

The H248MGW1 and H248TermGroupMGW1 configured settings are described in the following table:

Category	Settings
H248MGW1	
Simulated MGW	The simulated GW type is configured to Trunking Gateway and the controlling MGC is set to the H248MGC1 activity. H248TermGroupMGW1 is configured on the MGW activity.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to ServiceChange requests, Auto-reply to Audit requests, Auto-reply to Modify on Root termination, Use TransactionResponseAck , and Auto-register options are selected.
Profiles	The ETSI_TGW/1 profile is selected for the simulated MGW.
H248TermGroupMGW1	

Scenario	The test scenario channel comprises the script functions corresponding to the MGC transmitted commands . Media is exchanged bidirectionally by using a VoiceSession function.
Execution Settings	The corresponding scenario channel is configured to execute repeatedly during the test sustain time.
SDP	No overriding SDP settings are configured.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

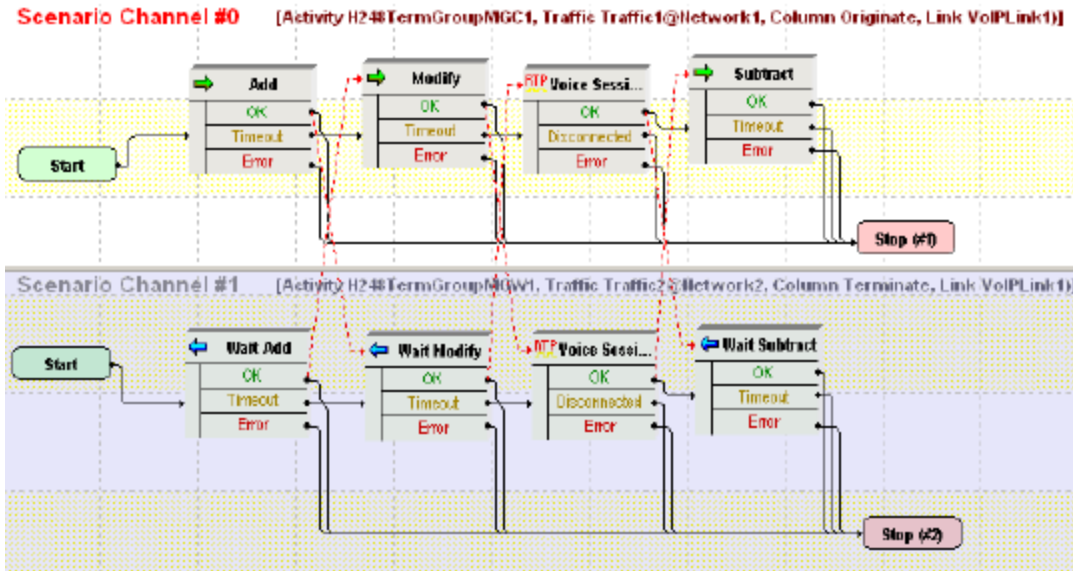
VM_021_H248_IPv4_B2B_without_registration

This test is similar to [VM_020_H248_IPv4_B2B_performance_topology1_rtp](#), except that no registration of the MGW is performed with the MGC. As such, the test does not have the **Wait for MGW Registration** (on the H248MGC1 activity) and the **Auto Register** (on the H248MGW1 activity) options configured.

VM_022_H248_IPv4_B2B_G711_ulaw

This test based on the configuration shown in T1 simulates a Trunking GW with a configured objective of 20 channels and a controlling MGC. The GW activity has an H248TermGroup defined that executes an H.248 signaling and media functions flow. The MGC has an associated H248TermGroup that executes an H248 and media functions flow.

The underlying two channel test scenario involving H248TermGroupMGC1 and H248TermGroupMGW1 is shown in the following image:



The H248MGC1 and H248TermGroupMGC1 configured settings are described in the following table:

Category	Settings
H248MGC1	
Simulated MGC	The controlled GW type is configured to Trunking Gateway (PSTN2IP) and the H248TermGroupMGC1 is enabled.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to ServiceChange requests, Auto-reply to Notify requests, Send Modify on Root termination to set properties, Use TransactionResponseAck , and Wait for MGW registration options are selected.
Profiles	The ETSI_TGW/1 profile is selected.
H248TermGroupMGC1	
Scenario	The test scenario comprises the Add and Modify messages followed by a media session that uses the VoiceSession script function. The terminations are eventually deleted by using a Subtract function.
Execution Settings	The corresponding scenario channel is configured to execute five times during the test sustain time.
SDP	No overriding SDP settings are configured.

Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

The H248MGW1 and H248TermGroupMGW1 configured settings are described in the following table:

Category	Settings
H248MGW1	
Simulated MGW	The simulated GW type is configured to Trunking Gateway and the controlling MGC is set to the H248MGC1 activity. H248TermGroupMGW1 is configured and enabled.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to ServiceChange requests, Auto-reply to Audit requests, Auto-reply to Modify on Root termination, Use TransactionResponseAck, and Auto-register options are selected.
Profiles	The ETSI_TGW/1 profile is selected for the simulated MGW.
H248TermGroupMGW1	
Scenario	The test scenario comprises Wait Add and Wait Modify, followed by a media session that uses the VoiceSession script function, and the Wait Subtract function.
Execution Settings	The corresponding scenario channel is configured to execute five times during the test sustain time.
SDP	No overriding SDP settings are configured.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

VM_023_H248_IPv4_B2B_G711_alaw

This test is similar to [VM_022_H248_IPv4_B2B_G711_ulaw](#), except that it uses the G711 alaw instead of the G711 ulaw codec.

VM_024_H248_IPv4_B2B_G723_1_5.3

This test is similar to [VM_022_H248_IPv4_B2B_G711_ulaw](#), except that it uses the G723 @5.3kbps instead of the G711 ulaw codec.

VM_025_H248_IPv4_B2B_G723_1_6.3

This test is similar to [VM_022_H248_IPv4_B2B_G711_ulaw](#), except that it uses the G723 @6.3kbps instead of the G711 ulaw codec.

VM_026_H248_IPv4_B2B_G726_16

This test is similar to [VM_022_H248_IPv4_B2B_G711_ulaw](#), except that it uses the G726 @16kbps instead of the G711 ulaw codec.

VM_027_H248_IPv4_B2B_G726_24

This test is similar to [VM_022_H248_IPv4_B2B_G711_ulaw](#), except that it uses the G726 @16kbps instead of the G711 ulaw codec.

VM_028_H248_IPv4_B2B_G726_32

This test is similar to [VM_022_H248_IPv4_B2B_G711_ulaw](#), except that it uses the G726 @32kbps instead of the G711 ulaw codec.

VM_029_H248_IPv4_B2B_G726_40

This test is similar to [VM_022_H248_IPv4_B2B_G711_ulaw](#), except that it uses the G726 @40kbps instead of the G711 ulaw codec.

VM_030_H248_IPv4_B2B_G729AB

This test is similar to [VM_022_H248_IPv4_B2B_G711_ulaw](#), except that it uses the G729AB instead of the G711 ulaw codec.

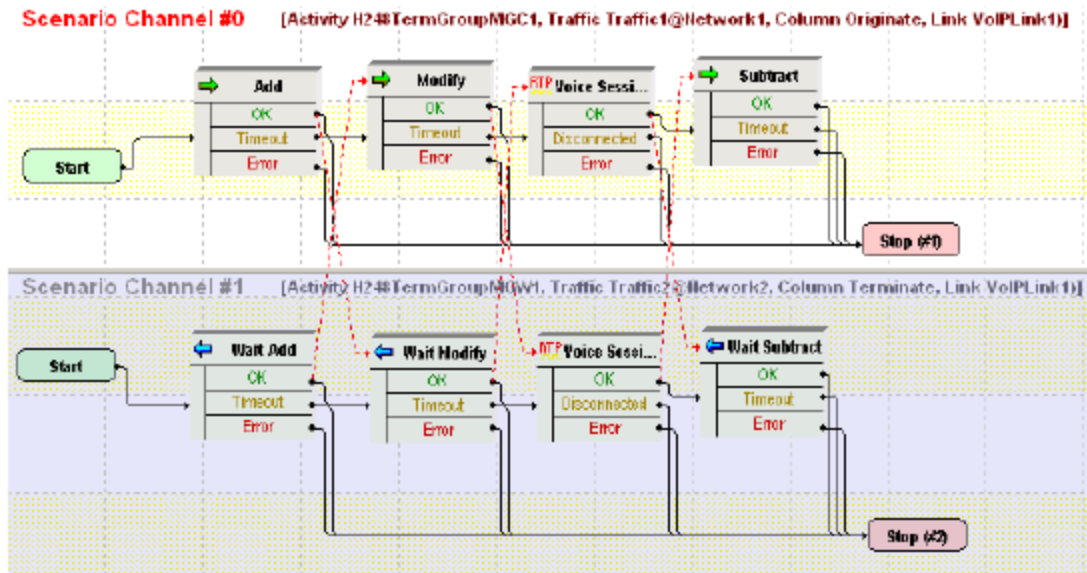
VM_031_H248_IPv4_B2B_ILBC

This test is similar to [VM_022_H248_IPv4_B2B_G711_ulaw](#), except that it uses the iLBC instead of the G711 ulaw codec.

VM_032_H248_IPv4_B2B_auto_sdp

This test based on the configuration shown in T1 simulates a Trunking GW with a configured objective of 20 channels and a controlling MGC. The GW activity has an H248TermGroup defined that executes an H.248 signaling and media functions flow. The MGC has an associated H248TermGroup that executes an H248 and media functions flow.

The underlying two channel test scenario involving H248TermGroupMGC1 and H248TermGroupMGW1 is shown in the following image:



The H248MGC1 and H248TermGroupMGC1 configured settings are described in the following table:

Category	Settings
H248MGC1	
Simulated MGC	The controlled GW type is configured to Trunking Gateway (PSTN2IP) and the H248TermGroupMGC1 is enabled.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to ServiceChange requests , Auto-reply to Notify requests , Send Modify on Root termination to set properties , Use TransactionResponseAck , and Wait for MGW registration options are selected.
Profiles	The ETSI_TGW/1 profile is selected.
H248TermGroupMGC1	
Scenario	The test scenario comprises the Add (localDescriptor = AutoSDP) and Modify (RemoteDescriptor = AutoSDP) messages followed by a media session that uses the VoiceSession script function. The terminations are eventually deleted by using a Subtract function.
Execution Settings	The corresponding scenario channel is configured to execute four times during the test sustain time.
SDP	No overriding SDP settings are configured.

Codec Settings	The G.711 u-law, G.711 u-law, G.726@16kbps, and G.726@24kbps codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

The H248MGW1 and H248TermGroupMGW1 configured settings are described in the following table:

Category	Settings
H248MGW1	
Simulated MGW	The simulated GW type is configured to Trunking Gateway and the controlling MGC is set to the H248MGC1 activity. H248TermGroupMGW1 is configured and enabled.
H248	Version 3 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto-reply to ServiceChange requests, Auto-reply to Audit requests, Auto-reply to Modify on Root termination, Use TransactionResponseAck, and Auto-register options are selected.
Profiles	The ETSI_TGW/1 profile is selected for the simulated MGW.
H248TermGroupMGW1	
Scenario	The test scenario comprises script functions matching the scenario channel #0 functions flow as shown in the preceding table.
Execution Settings	The corresponding scenario channel is configured to execute four times during the test sustain time.
SDP	No overriding SDP settings are configured.
Codec Settings	The G.711 u-law, G.711 u-law, G.726@16kbps, and G.726@24kbps codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

VM_033_H248_IPv4_B2B_custom_SDP

This test is similar to [VM_032_H248_IPv4_B2B_auto_sdp](#), except that it uses a custom SDP definition in the Modify script function on scenario channel #0.

VM_034_H248_IPv4_B2B_sdp_renegociation

This test is similar with [VM_033_H248_IPv4_B2B_custom_SDP](#), with the difference that at test scenario level, after the first media exchange session, the codec is re-negotiated by using another Modify function. Eventually, media is exchanged again by using a VoiceSession function.

VM_035_H248_IPv4_vs_DUT_RGW_analog_basic_call

This test simulates an MGC activity and runs against a Residential GW (DUT).

The H248MGC1 and H248TermGroupMGC1 configured settings are described in the following table:

Category	Settings
H248MGC1	
Simulated MGC	The controlled GW type is configured to Residential Gateway (PSTN2IP) and the H248TermGroupMGC1 is enabled. The IP address of the controlled GW is configured in the MGW column.
H248	Version 1 of the H.248/MEGACO protocol is configured.
Automatic Functionality	The Auto reply to ServiceChange requests and Wait for MGW registration options are selected.
Profiles	The ETSI_TGW/1 profile is selected.
H248TermGroupMGC1	
Scenario	The test scenario comprises an initial Modify for the physical termination, followed by Add and Modify functions for the RTP terminations. Media is exchanged by using the VoiceSession script function and the terminations are eventually deleted by using a Subtract function.
Execution Settings	The corresponding scenario channel is configured to execute five times during the test sustain time.
SDP	No overriding SDP settings are configured.
Codec Settings	The G.711 a-law and G.711 u-law codecs are selected.
RTP Settings	The Enable media on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

VM_036_H248_IPv4_vs_DUT_RGW_analog_basic_all_with_renegotiation

This test is similar to [VM_035_H248_IPv4_vs_DUT_RGW_analog_basic_call](#), except that codec renegotiation is done at scenario level by using Modify messages. After each negotiation, media is exchanged by using VoiceSession script functions.

VM_037_H248_IPv4_B2B_QoV

This test is similar to [VM_032_H248_IPv4_B2B_auto_sdp](#), except that QoV computation is selected in the RTP page of both H248TermGroup activities.

MGCP Sample Test Configuratins and Test Scenarios

This section describes the predefined IxLoad Voice Plug-in MGCP available sample test configurations (RXFs) and their associated test scenarios.

Note: For a complete description of the supported H.248 Test Library functions, see [VoIP MGCP Functions Library](#).

Used Test Configurations

All sample MGCP tests supplied with IxLoad are based on one of the following three configurations (C1, C2, and C3):

- **GW/GW:** IxLoad is used to simulate two GWs, while the CA is a real test device (DUT). The combined MGCP protocol and RTP functions flow is mapped to Endpoints that are associated with each of the MGCP GW activities. See configuration C1 in the following image:



- **GW/CA:** IxLoad is used to simulate a GW and its controlling CA. In this test configuration, the CA is intended to simulate an additional GW that is controlled by the CA and whose endpoints perform RTP media exchange with the MGCPGW1 endpoints. The combined MGCP protocol and RTP functions flow is mapped to Endpoints that are associated with each of the MGCP GW and MGCP CA activities. See configuration C2 in the following image:



- **Two GWs/CA:** IxLoad is used to simulate two GWs and their controlling CA. The combined MGCP protocol and RTP functions flow is mapped to Endpoints that are associated with each of the MGCPGW1 and MGCPGW2 activities. Note that the MGCPA1 activity is configured with two Endpoint activities, each one corresponding to a controlled GW, each one executing an MGCP-

only protocol flow. See configuration C3 in the following image:



VoIP MGCP Test Configurations

The following sample VoIP MGCP test configuration files are contained in the IxLoad installer:

VMG_001_MGCP_IPV4_B2B_Basic_Call_Gw_vs_CA_with_RTP

This test based on the configuration shown in C2 runs in B2B mode and simulates MGCPGW1 endpoints calling other endpoints, with media streaming performed after call setup.

Both the GW and the CA activity have Endpoints defined that execute an MGCP signaling and RTP media functions flow.

The MGCPGW1 and Endpoint1 configured settings are described in the following table:

Category	Settings
MGCPGW1	
Simulated Gateways	A number of 100 gateways (gw[001-100]) are configured with a single endpoint each. The controlling CA is MGCPA1.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint1	
Scenario	The scenario channel implements a simple call sequence with media streaming performed after call setup.
Execution Settings	The corresponding scenario channel is configured to execute two loops during the test sustain time.
Simulated Endpoints	The destination phone number is specified by using the 170[000000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.

RTP	RTP hardware acceleration is configured.
Audio	The Enable audio on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

The MGCPGA1 and Endpoint2 configured settings are described in the following table:

Category	Settings
MGCPGA1	
Simulated Gateways	The controlled gateways are specified as one set with a number of 100 gateways (gw [001-100]) with an endpoint each, corresponding to MGCPGW1.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint2	
Scenario	The scenario channel implements a simple call sequence (receiving side) with media streaming performed after call setup.
Execution Settings	The corresponding scenario channel is configured to execute two loops during the test sustain time.
Simulated Endpoints	The phone number is specified by using the 170[000000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	RTP hardware acceleration is configured.
Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

VMG_002_MGCP_IPV4_B2B_Basic_Call_Gw_vs_CA_signalling_only

This test is similar to [VMG_001_MGCP_IPV4_B2B_Basic_Call_Gw_vs_CA_with_RTP](#), with the only difference that the call flow is signaling-only (no Voice Session functions).

VMG_003_MGCP_IPV4_B2B_Basic_Call_GW1_calls_GW2_through_CA

This test based on the configuration shown in C3 test runs in B2B mode and simulates GW1 endpoints calling endpoints provisioned on another GW through a CA, with media streaming performed after call setup.

Both GWs and the CA activity have Endpoints defined that execute an MGCP signaling flow (between the GW and CA) and RTP media functions flow (between endpoints on GWs).

The MGCPGW1 and Endpoint1 configured settings are described in the following table:

Category	Settings
MGCPGW1	
Simulated Gateways	A number of 8000 gateways (gw[00000-07999]) are configured with a single endpoint each. The controlling CA is MGCPCA1.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint1	
Scenario	Endpoints perform an initialization procedure and initiate the call by exchanging MGCP messages with the CA, after which media streaming is performed by using the VoiceSession script function. Eventually, the endpoints terminate the call following a request from the CA.
Execution Settings	The corresponding scenario channel is configured to execute two loops during the test sustain time.
Simulated Endpoints	The destination phone number is specified by using the 170[000000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	No RTP settings are configured.
Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

Note: The scenario-level configuration for MGCPGW2 and Endpoint3 is similar to the previous one, with the difference that the Endpoint3-simulated endpoints initiate the call termination. At activity-level, MGCPGW2 is configured to simulate number of 8000 gateways (gw[08000-15999]) with a single endpoint each. On Endpoint 3, the call destination is configured as 160 [00000-], but it is not used by the test, because Endpoint3 only receives a call.

The MGCPA1 and Endpoint2 configured settings are described in the following table:

Category	Settings
MGCPGCA1	
Simulated Call Agent	The controlled gateways are specified as two sets, one with a number of 8000 gateways corresponding to MGCPGW1 and another set of 8000 gateways corresponding to MGCPGW2.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint2	
Scenario	The test scenario implements the CA-side MGCP-only message exchange with the endpoints simulated by MGCPGW1.
Execution Settings	The corresponding scenario channel is configured to execute two loops during the test sustain time.
Simulated Endpoints	The source phone number for MGCPGW1 endpoints is specified by using the 160 [000000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	No RTP settings are configured.
Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

Note: The Endpoint4 activity implements the CA-side MGCP-only message exchange with MGCPGW2 simulated endpoints. Endpoint4 is configured similar to Endpoint2, with the difference that it specifies the source phone numbers using the 170[00000-] sequence generator expression.

VMG_004_MGCP_IPV4_B2B_Basic_Call_with_Renegociation

This test based on the configuration shown in C2 runs in B2B mode and simulates MGCPGW1 endpoints calling other endpoints, with media streaming performed after call setup. Following a first media streaming session, codec renegotiation occurs (MDCX) and another media streaming session is performed.

Both the GW and the CA activity configured by this have Endpoint activities associated defined that execute an MGCP signaling and RTP media functions flow.

The MGCPGW1 and Endpoint1 configured settings are described in the following table:

Category	Settings
MGCPGW1	
Simulated Gateways	A number of 100 gateways (gw[001-100]) are configured with a single endpoint each. The controlling CA is MGCPA1.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint1	
Scenario	The test scenario comprises the GW Make Call procedure, Voice Session function, Wait MDCX, Voice Session, and GW End Call procedure.
Execution Settings	The corresponding scenario channel is configured to execute one loop during the test sustain time.
Simulated Endpoints	The destination phone number is specified by using the 170[000000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	RTP hardware acceleration is configured.
Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

The MGCPA1 and Endpoint2 configured settings are described in the following table:

Category	Settings
MGCPGW2	
Simulated Call Agent	The controlled gateways are specified as a set of 100 gateways (gw[001-100]) corresponding to MGCPGW1.

Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint2	
Scenario	The test scenario mirrors the Endpoint1 call flow for the receiving side and comprises the Wait ReceiveCall, Voice Session function, Send MDCX, Voice Session, and Wait Recv_EndCall.
Execution Settings	The corresponding scenario channel is configured to execute one loop during the test sustain time.
Simulated Endpoints	The endpoint phone number is specified by using the 170[000000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	RTP hardware acceleration is configured.
Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

VMG_005_MGCP_IPV4_B2B_Basic_Call_with_DTMFs

This test is similar to [VMG_001_MGCP_IPV4_B2B_Basic_Call_Gw_vs_CA_with_RTP](#), with the main difference that the underlying test scenario contains Generate DTMF/Detect DTMF functions instead of the Voice Session functions.

The test runs in B2B mode and simulates MGCPGW1 endpoints calling other endpoints, with DTMF sending/receiving performed after call setup.

Both the GW and the CA activity have Endpoint activities defined that execute an MGCP signaling and a media functions flow.

The MGCPGW1 and Endpoint1 configured settings are described in the following table:

Category	Settings
MGCPGW1	
Simulated Gateways	A number of 100 gateways (gw[001-100]) are configured with a single endpoint each. The controlling CA is MGCPCA1.

Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint1	
Scenario	The test scenario comprises the GW Make Call procedure, Generate DTMF function, and GW End Call procedure.
Execution Settings	The corresponding scenario channel is configured to execute two loops during the test sustain time.
Simulated Endpoints	The destination phone number is specified by using the 170[000000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	RTP hardware acceleration is configured.
Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

The MGCPA1 and Endpoint2 configured settings are described in the following table:

Category	Settings
MGCPGW2	
Simulated Call Agent	The controlled gateways are specified as a set of 100 gateways (gw[001-100]) corresponding to MGCPGW1.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint2	
Scenario	The test scenario mirrors the Endpoint1 call flow for the receiving side and comprises the Wait ReceiveCall, Receive DTMF, and Wait Recv_EndCall.
Execution Settings	The corresponding scenario channel is configured to execute two loops during the test sustain time.
Simulated Endpoints	The endpoint phone number is specified by using the 170[000000-] sequence generator expression.

SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	RTP hardware acceleration is configured.
Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

VMG_006_MGCP_IPV4_B2B_Basic_Call_with_voice_session_G711_Alaw

This test based on the configuration shown in C2 runs in B2B mode and simulates MGCPGW1 endpoints calling other endpoints, with media streaming performed after call setup.

Both the GW and the CA activity configured by this test have Endpoint activities associated defined that execute an MGCP signaling and RTP media functions flow.

The MGCPGW1 and Endpoint1 configured settings are described in the following table:

Category	Settings
MGCPGW1	
Simulated Gateways	A number of 100 gateways (gw[001-100]) are configured with a single endpoint each. The controlling CA is MGCPCA1.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint1	
Scenario	The test scenario implements a common call setup sequence (caller) with media exchange following call establishment. It comprises the GW Make Call procedure, Voice Session function, and GW End Call procedure.
Execution Settings	The corresponding scenario channel is configured to execute two loops during the test sustain time.
Simulated Endpoints	The destination phone number is specified by using the 170[000000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	RTP hardware acceleration is configured.

Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

The MGCPA1 and Endpoint2 configured settings are described in the following table:

Category	Settings
MGCPGW2	
Simulated Call Agent	The controlled gateways are specified as a set of 100 gateways (gw[001-100]) corresponding to MGCPGW1.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint2	
Scenario	The test scenario mirrors the Endpoint1 call flow for the receiving side (called party) and comprises the Wait ReceiveCall, Voice Session, and Wait Recv_EndCall.
Execution Settings	The corresponding scenario channel is configured to execute two loops during the test sustain time.
Simulated Endpoints	The endpoint phone number is specified by using the 170[000000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	RTP hardware acceleration is configured.
Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

VMG_007_MGCP_IPV4_B2B_Basic_Call_with_voice_session_G726-40

This test is similar to that [VMG_006_MGCP_IPV4_B2B_Basic_Call_with_voice_session_G711_Alaw](#), with the only difference that RTP streaming uses the G726@40kbps codec, configured in the **Codecs** tab of each Endpoint activity.

VMG_008_MGCP_IPV4_B2B_Basic_Call_with_RSIP_from_scenario

This test based on the configuration shown in C2 runs in B2B mode and simulates MGCPGW1 endpoints calling other endpoints, with media streaming performed after call setup.

The main difference to [VMG_009_MGCP_IPV4_B2B_Basic_Call_with_strip_leading_zero_enabled](#) is that at test scenario level MGCPGW1 endpoints are configured to send an MGCP RSIP message in the Init_Endpoint procedure that is executed before initiating the call setup sequence. Correspondingly, the CAside message flow on the Endpoint2 activity contains in the Init_Endpoint procedure a Wait RSIP script function that handles the incoming RSIP message.

VMG_009_MGCP_IPV4_B2B_Basic_Call_with_strip_leading_zero_enabled

This test based on the configuration shown in C2 runs in B2B mode and simulates MGCPGW1 endpoints calling other endpoints, with RTP media streaming performed after call setup.

Both the GW and the CA activity configured by this test have Endpoint activities associated defined that execute an MGCP signaling and an RTP media functions flow.

The MGCPGW1 and Endpoint1 configured settings are described in the following table:

Category	Settings
MGCPGW1	
Simulated Gateways	A number of 100 simulated gateways (gw[001-100]) are configured with 900 endpoints (aaln[001-900]) each. The controlling CA is MGCPA1. The Strip leading zeros from endpoint name option is enabled.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint1	
Scenario	The test scenario implements a common call setup sequence (caller) with media exchange following call establishment. It comprises the GW Make Call procedure, Voice Session function, and GW End Call procedure.
Execution Settings	The corresponding scenario channel is configured to execute 10 loops during the test sustain time.
Simulated Endpoints	The destination phone number is specified by using the 170[000000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	RTP hardware acceleration is configured.
Audio	The Enable audio on this activity option is selected.

SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

The MGCPA1 and Endpoint2 configured settings are described in the following table:

Category	Settings
MGCPGW2	
Simulated Call Agent	The number of controlled gateways is specified one set of 100 gateways with 900 endpoints (aaln[001-900]) each, corresponding to MGCPGW1.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint2	
Scenario	The test scenario mirrors the Endpoint1 call flow for the receiving side (called party) and comprises the Wait ReceiveCall, Voice Session, and Wait Recv_EndCall.
Execution Settings	The corresponding scenario channel is configured to execute 10 loops during the test sustain time.
Simulated Endpoints	The endpoint phone number is specified by using the 170[000000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	RTP hardware acceleration is configured.
Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

VMG_010_MGCP_IPV4_VS_DUT_BTS_Basic_Call_with_BTS

This test based on the configuration shown in C1 runs against a real Call Agent device (BTS) and simulates MGCPGW1 endpoints calling other endpoints through the CA, with media streaming performed between endpoints after call setup.

Each MGW has an Endpoint activity defined that executes the MGCP flow with the CA and the media streaming towards the opposite MGW.

The MGCPGW1 and Endpoint1 configured settings are described in the following table:

Category	Settings
MGCPGW1	
Simulated Gateways	A number of 100 simulated gateways (ix[8002-8101]) are configured with a single endpoint each. The controlling CA is the DUT, which is specified by its IP address.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint1	
Scenario	MGCPGW1 endpoints call the endpoints on MGCPGW2 through the CA and perform media streaming (VoiceSession script function) over the established call.
Execution Settings	The corresponding scenario channel is configured to execute 10 loops during the test sustain time.
Simulated Endpoints	The destination phone number is specified by using a sequence generator expression that generates 100 numbers.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	No special RTP settings are configured.
Audio	The Enable audio on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

The MGCPGW2 and Endpoint2 configured settings are described in the following table:

Category	Settings
MGCPGW2	
Simulated Gateways	A number of 100 simulated gateways (ix[8102-8201]) are configured with a single endpoint each. The controlling CA is the DUT, is specified by its IP address.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint2	
Scenario	MGCPGW2 endpoints exchange MGCP messages with the CA for receiving the incoming call, and then stream RTP media (VoiceSession script function) over the established call.

Execution Settings	The corresponding scenario channel is configured to execute repeatedly for the test sustain time.
Simulated Endpoints	No destination phone number is specified.
SDP	No overriding SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	No special RTP settings are configured.
Audio	The Enable audio on this activity option is selected.
Other Settings	No scenario variables need to be initialized.

VMG_011_MGCP_IPV4_B2B_HWRTP_Non_Agg_Basic_Call_RTP_8000ch

This test based on the configuration shown in C2 runs in B2B mode and simulates endpoints calling other endpoints, with RTP media streaming performed after call setup.

Both the GW and the CA activity configured by this test have Endpoint activities associated defined that execute an MGCP signaling and an RTP media functions flow.

The MGCPGW1 and Endpoint1 configured settings are described in the following table:

Category	Settings
MGCPGW1	
Simulated Gateways	A number of 8000 gateways (gw[0001-8000]) are configured with a single endpoint each. The controlling CA is MGCPCA1.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint1	
Scenario	The test scenario implements a common call setup sequence (caller) with media exchange following call establishment. It comprises the GW Make Call procedure, Voice Session function, and GW End Call procedure.
Execution Settings	The corresponding scenario channel is configured to execute two loops during the test sustain time.
Simulated Endpoints	The destination phone number is specified by using the 160[000000-] sequence generator expression.
SDP	No custom SDP settings are configured.

Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	RTP hardware acceleration is configured.
Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

The MGCPA1 and Endpoint2 configured settings are described in the following table:

Category	Settings
MGCPGW2	
Simulated Call Agent	The number of controlled gateways is specified one set of 8000 gateways corresponding to MGCPGW1.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint2	
Scenario	The test scenario mirrors the Endpoint1 call flow for the receiving side (called party) and comprises the Wait ReceiveCall, Voice Session, and Wait Recv_EndCall.
Execution Settings	The corresponding scenario channel is configured to execute two loops during the test sustain time.
Simulated Endpoints	The endpoint phone number is specified by using the 160[00000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	RTP hardware acceleration is configured.
Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

VMG_012_MGCP_IPV4_B2B_HWRTP_1G_Agg_Basic_Call_RTP_8000ch

This test is similar to [VMG_011_MGCP_IPV4_B2B_HWRTP_Non_Agg_Basic_Call_RTP_8000ch](#), with the only difference that 1 GB traffic aggregation for Acceleron load modules is used.

VMG_013_MGCP_IPV4_B2B_HWRTP_10G_Agg_Basic_Call_RTP_96000ch

This test based on the configuration shown in C2 runs in B2B mode and simulates GW endpoints calling other endpoints, with RTP media streaming performed after call setup.

Both the GW and the CA activity configured by this test have Endpoint activities associated defined that execute an MGCP signaling and an RTP media functions flow.

The test has an associated test objective of 96000 channels and has 10 Gb traffic aggregation for Acceleron load modules configured.

The MGCPGW1 and Endpoint1 configured settings are described in the following table:

Category	Settings
MGCPGW1	
Simulated Gateways	A number of 10 gateways (gw[01-12]) are configured with a number of 8000 (aaln [0001-8000]) endpoints each. The controlling CA is MGCPCA1.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint1	
Scenario	The test scenario implements a common call setup sequence (caller) with media exchange following call establishment. It comprises the GW Make Call procedure, Voice Session function, and GW End Call procedure.
Execution Settings	The corresponding scenario channel is configured to execute two loops during the test sustain time.
Simulated Endpoints	The destination phone number is specified by using the 160[000000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	RTP hardware acceleration is configured.
Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

The MGCPA1 and Endpoint2 configured settings are described in the following table:

Category	Settings
MGCPGW2	
Simulated Call Agent	The number of controlled gateways is specified one set of 12 gateways with 8000 endpoints each, corresponding to MGCPGW1.
Automatic Functionality	The sending of RSIP messages at the beginning and the end of the test is configured, and the Enable retransmissions option is selected.
Endpoint2	
Scenario	The test scenario mirrors the Endpoint1 call flow for the receiving side (called party) and comprises the Wait ReceiveCall, Voice Session, and Wait Recv_EndCall.
Execution Settings	The corresponding scenario channel is configured to execute two loops during the test sustain time.
Simulated Endpoints	The endpoint phone number is specified by using the 160[00000-] sequence generator expression.
SDP	No custom SDP settings are configured.
Codecs	The G.711 a-law and G.711 u-law codecs are selected.
RTP	RTP hardware acceleration is configured.
Audio	The Enable audio on this activity option is selected.
SRTP	Use of SRTP is not enabled.
Other Settings	No scenario variables need to be initialized.

PSTN Sample Test Configuratins and Test Scenarios

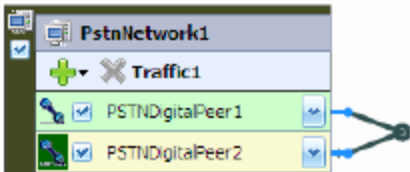
This section describes the predefined IxLoad Voice Plug-in PSTN sample test configurations (RXFs) and their associated test scenarios.

Note: For a complete description of the supported digital T1/E1 Test Library functions, see [Digital T1/E1 Functions Library on page 160](#).

Used Test Configurations

All sample PSTN tests supplied with IxLoad are based on one of the following two configurations (C1 and C2):

- T1/E1 Digital only tests: The PSTN-only test flow is generated by two PSTNDigitalPeer activities that exchange digital T1/E1 signaling and media traffic with each other. Both PSTNDigitalPeer activities are configured on the same PSTN NetTraffic.



Tests from this category can run either in back-to-back mode (these have a B2B string in their name), or against a DUT (VS string in their name), such as a router or voice gateway.

- Mixed SIP/Digital T1/E1 tests: The SIP and Digital T1/E1 test flow is exchanged between VoIPSPeer and PSTNDigitalPeer activities that establish between themselves calls with media exchange.



PSTN Test Configurations

The following sample PSTN test configuration files are contained in the IxLoad installer:

PSTN_001_B2B_T1_CAS_IMM_D4_B8ZS

This test, which is based on the configuration shown in C1, illustrates the case of calls initiated by the PSTNDigitalPeer1 activity and terminated by the PSTNDigitalPeer2 activity. After successful call setup, bidirectional media is exchanged between the call participants by using the Voice Session script function.

The network-level configured settings are T1 CAS, D4 framing and B8SZ line encoding, Immediate variant for both PSTN digital ranges.

The activity-level PSTNDigitalPeer1 and PSTNDigitalPeer2 configured settings are described in the following table:

Category	Settings
PSTNDigitalPeer1	
Scenario	The associated scenario channel implements a simple call sequence (Make Call, Voice Session, End Call) with bidirectional media streaming performed after call setup.
Execution Settings	The corresponding scenario channel is configured to execute five loops during the test sustain time.
Dial Plan	The PSTNDigitalPeer source phone numbers are specified by using a 150[00000000-] sequence generator expression. The activity is configured to initiate a call to the PSTNDigitalPeer2 activity specified by using a symbolic link.
Audio	PSTNDigitalPeer media functions are configured to play a specified audio clip for its entire duration.
PSTNDigitalPeer2	
Scenario	The associated scenario channel implements a simple call sequence for the terminating side (Receive Call, Voice Session, End Call) with media streaming performed after call setup.
Execution Settings	The corresponding scenario channel is configured to execute five loops during the test sustain time.
Dial Plan	Because this activity only terminates the call, no call destination is configured.
Audio	PSTNDigitalPeer media functions are configured to play a specified audio clip for its entire duration.

PSTN_002_B2B_T1_CAS_FGD_D4_B8ZS

This test based on the configuration shown in C1 runs in B2B mode. This test is similar to the previous [PSTN_001_B2B_T1_CAS_IMM_D4_B8ZS](#) one, with the difference that PSTNDigitalPeer activities are configured by using the FGD instead of the Immediate variant at network level.

PSTN_003_B2B_E1_CAS_Argentina_G704_HDB3

This test based on the configuration shown in C1 runs in B2B mode. This test is similar to the previous [PSTN_001_B2B_T1_CAS_IMM_D4_B8ZS](#) one, with the difference that PSTNDigitalPeer activities are configured by using the E1 type, CAS signaling, G704 framing, and HDB3 line encoding at network level.

PSTN_004_B2B_T1_4ESS_D4_B8ZS

This test based on the configuration shown in C1 runs in B2B mode. This test is similar to the previous [PSTN_001_B2B_T1_CAS_IMM_D4_B8ZS](#) one, with the difference that PSTNDigitalPeer activities are configured by using the T1 type, ISDN PRI signaling, E4 framing, B8ZS line encoding, and 4ESS protocol at network level.

PSTN_005_B2B_T1_QSIG_D4_B8ZS

This test based on the configuration shown in C1 runs in B2B mode. This test is similar to the previous [PSTN_001_B2B_T1_CAS_IMM_D4_B8ZS](#) one, with the difference that PSTNDigitalPeer activities are configured by using the T1 type, ISDN PRI signaling, D4 framing, B8ZS line encoding, and QSIG protocol at network level.

PSTN_006_B2B_T1_5ESS_D4_B8ZS

This test based on the configuration shown in C1 runs in B2B mode. This test is similar to the previous [PSTN_001_B2B_T1_CAS_IMM_D4_B8ZS](#) one, with the difference that PSTNDigitalPeer activities are configured by using the T1 type, ISDN PRI signaling, D4 framing, B8ZS line encoding, and 5ESS protocol at network level.

PSTN_007_B2B_E1_ISDN_QSIG_G704_HDB3

This test based on the configuration shown in C1 runs in B2B mode. This test is similar to the previous [PSTN_001_B2B_T1_CAS_IMM_D4_B8ZS](#) one, with the difference that PSTNDigitalPeer activities are configured by using the E1 type, ISDN PRI signaling, G704 framing, HDB3 line encoding, and QSIG protocol at network level.

PSTN_008_B2B_E1_ISDN_KHT_G704_HDB3

This test based on the configuration shown in C1 runs in B2B mode. This test is similar to the previous [PSTN_001_B2B_T1_CAS_IMM_D4_B8ZS](#) one, with the difference that PSTNDigitalPeer activities are configured by using the E1 type, ISDN PRI signaling, G704 framing, HDB3 line encoding, and KHT protocol at network level.


PSTN_009_VS_Cisco_E1_ISDN_vs_E1_ISDN

This test, which is based on the configuration shown in C2, illustrates the case of calls initiated by the PSTNDigitalPeer1 activity and terminated by the PSTNDigitalPeer2 activity. After successful call setup, bidirectional media is exchanged between the call participants by using the Voice Session script function.

The network-level configured settings are E1 ISDN PRI, G704 framing HDB3 line encoding, QSIG protocol for both PSTN digital ranges.

The PSTNDigitalPeer1 and PSTNDigitalPeer2 configured settings are described in the following table:

Category	Settings
PSTNDigitalPeer1	
Scenario	The associated scenario channel implements a simple call sequence (Make Call, Voice Session, End Call) with bidirectional media streaming performed after call setup.

Execution Settings	The corresponding scenario channel is configured to execute five loops during the test sustain time.
Dial Plan	The PSTNDigitalPeer1 source phone numbers are specified by using a 150[00000000-] sequence generator expression. The activity is configured to initiate calls to the PSTNDigitalPeer2 activity specified by using a symbolic link.
Audio	PSTNDigitalPeer1 media functions are configured to play a specified audio clip for its entire duration.
PSTNDigitalPeer2	
Scenario	The associated scenario channel implements a simple call sequence for the terminating side (Receive Call, Voice Session, End Call) with bidirectional media streaming performed after call setup.
Execution Settings	The corresponding scenario channel is configured to execute five loops during the test sustain time.
Dial Plan	Because this activity only terminates the call, no call destination is configured. The source phone numbers are specified by using the 20[00-] sequence generator expression.  Note: This sequence must be present on the DUT the test is run against.
Audio	PSTNDigitalPeer1 media functions are configured to play a specified audio clip for its entire duration.

PSTN_010_VS_Cisco_SIP_vs_T1

This mixed SIP - Digital T1/E1 test, which is based on the configuration shown in C2, illustrates the case of calls initiated by the VoIPSIPPeer1 activity and terminated by the PSTNDigitalPeer1 activity. After successful call setup, bidirectional media is exchanged between the call participants by using the Voice Session script function.

The network-level PSTNDigitalPeer1 configured settings are T1 ISDN PRI, 4ESS protocol, ESF framing, B8ZS line encoding for the network range pertaining to the activity.

The activity-level VoIPSIPPeer1 and PSTNDigitalPeer1 configured settings are described in the following table:

Category	Settings
VoIPSIPPeer1	
Scenario	The associated scenario channel implements a simple call sequence (Make Call, Voice Session, End Call) with bidirectional media streaming performed after call setup.
Execution Settings	The corresponding scenario channel is configured to execute one loop during the test sustain time.

Dial Plan	The source phone number is specified by using the 7777[00-] sequence generator expression. The IP address of the DUT is configured in the destination Destination IP field. The destination phone numbers are specified by using the 9999[00-] expression and the Override phone numbers from destination activity option is selected.
SIP	The SIP UAs emulated by this activity are configured to use an outbound SIP proxy server specified by using an IP address.
Automatic	Retransmissions and timers are not configured.
Codecs	The default G.711 a-law and G.711 u-law codecs are selected.
RTP	No custom RTP settings are configured.
Audio	The Enable audio on this activity option is selected.
Other	No scenario variables need to be initialized.
PSTNDigitalPeer1	
Scenario	The associated scenario channel implements a simple call sequence (Receive Call, Voice Session, End Call) with bidirectional media streaming performed after call setup.
Execution Settings	The corresponding scenario channel is configured to execute five loops during the test sustain time.
Dial Plan	The PSTNDigitalPeer1 source phone numbers are specified by using a 9999[00-] sequence generator expression.
Audio	PSTNDigitalPeer1 media functions are configured to play a specified audio clip for its entire duration.

PSTN_011_VS_Cisco_T1_CAS_IMM_vs_T1_CAS_IMM

This test based on the configuration shown in C1 runs against a DUT. This test is similar to the previous [PSTN_009_VS_Cisco_E1_ISDN_vs_E1_ISDN](#) one, with the difference that PSTNDigital activities are configured by using T1, CAS signaling, ESF framing, B8ZS line encoding, and E&M protocol at network level.

PSTN_012_VS_Cisco_T1_ISDN_5ESS_vs_T1_ISDN_5ESS

This test is similar to the previous [PSTN_009_VS_Cisco_E1_ISDN_vs_E1_ISDN](#) one, with the difference that PSTNDigital activities are configured using T1 type, ISDN PRI signaling, ESF framing, B8ZS line encoding, and 5ESS protocol at network level.

PSTN_013_VS_Cisco_T1_vs_SIP

This mixed SIP - Digital T1/E1 test, which is based on the configuration shown in C2, illustrates the case of calls initiated by the PSTNDigitalPeer1 activity and terminated by the VoIPSIPPeer1 activity. After successful call setup, bidirectional media is exchanged between the call participants by using the Voice Session script function.

The network-level PSTNDigitalPeer1 configured settings are T1 ISDN PRI, 4ESS protocol, ESF framing, and B8ZS line encoding for the network range pertaining to the activity.

The activity-level VoIPSIPPeer1 and PSTNDigitalPeer1 configured settings are described in the following table:

Category	Settings
PSTNDigitalPeer1	
Scenario	The associated scenario channel implements a simple call sequence (Make Call, Voice Session, End Call) with bidirectional media streaming performed after call setup.
Execution Settings	The corresponding scenario channel is configured to execute five loops during the test sustain time.
Dial Plan	The PSTNDigitalPeer1 source phone numbers are specified by using a 9999[00-] sequence generator expression. VoIPSIPPeer1 is configured as destination activity. The destination phone numbers are specified by using the 7777[00-] expression, and the Override phone numbers from destination activity option is selected.
Audio	PSTNDigitalPeer1 media functions are configured to play a specified audio clip for its entire duration.
VoIPSIPPeer1	
Scenario	The associated scenario channel implements a simple call sequence (Receive Call, Voice Session, End Call) with bidirectional media streaming performed after call setup.
Execution Settings	The corresponding scenario channel is configured to execute one loop during the test sustain time.
Dial Plan	The source phone number is specified using the 7777[00-] sequence generator expression.
SIP	The SIP UAs emulated by this activity are configured to use an outbound SIP proxy server specified by using an IP address.
Automatic	Retransmissions and timers are not configured.
Codecs	The default G.711 a-law and G.711 u-law codecs are selected.
RTP	No custom RTP settings are configured.
Audio	The Enable audio on this activity option is selected.
Other	No scenario variables need to be initialized.

CHAPTER 5 Extended Functionality SIP Tests Suite

This chapter covers the following topics:

- [SIP Tests Suite Overview](#)
- [Predefined Common Test Procedures](#)
- [Test Descriptions](#)

SIP Tests Suite Overview

The Extended Functionality SIP test suite consists of a set of test configurations (IxLoad rxf and tst files) aimed at covering a broad range of SIP device or VoIP network testing needs. The test suite is able to cover variations of the SIP implementations (different types of DUT) by using parameters in the test configurations.

Engineers testing SIP implementations use specific test plans focused on the area of interest. Implementing the test plan in the IxLoad configuration—and more generally, in any advanced test tool—requires a considerable effort especially for VoIP functionality tests. Having access to a comprehensive set of configuration reduces the time spent in building configurations and allows testers to focus on the DUT, and not on the test tool.

The customers can use the test suite by selecting the subset of tests that match their specific test plan, changing the specific, DUT-related parameters, and execute tests by using the IxLoad Voice Plug-in module. Although in some cases, a customization of the call flow and SIP message parameters would still be needed, this will be much easier than the approach whereby the configurations have to be created based on the supplied set of samples.

General Test Features

The supplied test configurations share the following specifications:

General Settings

- The configurations (rx) are provided for Acceleron-XP load module cards, having one chassis port allocated to each VoIPSIP activity.
- For all configurations, a single test scenario per configuration is used, whereby the name of the test scenario is the same as that of the configuration, but with a different extension.

Network Settings

The values of the test parameters—IP addresses for NetTraffics, the IP address of the SIP Proxy/Registrar server, the dial plans, the domain names—are consistent across the configurations. The used network addresses values are 20.1.1.1/16 for the first VoIPSIP activity, 20.1.50.1 for the second, and so on, 20.1.254.254 for the SIP Proxy/Registrar IP address, and ixload-test.com for domain name.

Generally, tests configured with the Use consecutive values (per port) IP address allocation scheme configured in the VoIPSIP activity's Execution page have 8000 IP addresses defined (starting from 20.1.1.1/16), while those that use the Use the same value (per port) IP address allocation scheme have 12 IP addresses defined (starting from 20.1.1.1/16).

To prevent the flooding of testbed switches, the sending of Gratuitous ARP messages is disabled, and ARP requests are made at test execution time for the emulated SIP UAs only.

Activity Settings

- The test configurations with media exchange (RTP) have audio (voice) full duplex support using the G.711 codec, with hardware acceleration enabled. For tests with the LPS objective type that use media script functions, these are configured to play media for the duration of the TalkTime

call parameter.

- Generally, tests configured by using an AC or LPS objective have the Use consecutive values (per port) IP address allocation scheme configured in the VoIPSIP activity's Execution page.

The image shows two side-by-side configuration windows. The left window is titled 'Channel mapping rules for SIP UA' and contains three dropdown menus: 'IP address:' set to 'Use consecutive values (per port)', 'UDP/TCP/TLS port:' set to 'Use same value', and 'Phone no:' set to 'Use consecutive values (per activity)'. There is a checked checkbox at the bottom labeled 'Accept multiple channels sharing the same IP:Port'. The right window is titled 'Channel mapping rules for media' and contains two dropdown menus: 'IP address:' set to 'Use consecutive values (per port)' and 'Port:' set to 'Use same value'.

- Test configured with a Channels objective use multiple SIP UAs that share the same IP address (the Channel mapping for SIP UAs parameter in the VoIPSIP activity's Execution page is set to Use same value (per port) value). The media address is also shared by all UAs (the Channel mapping for media parameter is set to Use same value (per port) value), while the media port uses consecutive values. These tests are identified by the MCH suffix appended to the test name.

The image shows two side-by-side configuration windows. The left window is titled 'Channel mapping rules for SIP UA' and contains three dropdown menus: 'IP address:' set to 'Use same value (per port)', 'UDP/TCP/TLS port:' set to 'Use same value', and 'Phone no:' set to 'Use consecutive values (per port)'. There is a checked checkbox at the bottom labeled 'Accept multiple channels sharing the same IP:Port'. The right window is titled 'Channel mapping rules for media' and contains two dropdown menus: 'IP address:' set to 'Use same value (per port)' and 'Port:' set to 'Use consecutive values (per port)'.

- For tests that have an AC or LPS objective type, media functions used in the test scenario are configured for playing audio/video media for the duration of the TalkTime call parameter (Play for clip duration or TalkTime option in the Audio page). For tests that use a Channels objective type, media functions are configured for playing audio/video media for a specified duration of time (seconds).

Scenario Settings

- All call flows include a SIP REGISTER message that is executed only once, on the first iteration of the test. Session timers for the Register operation are enabled at activity level in the VoIPSIP activity's Automatic page (the Expires header is configured to 3600 seconds, re-registrations are handled automatically every 3500 seconds).
- Most of the call flows are created so as to support the SIP Route/Record-Route mechanism. The use of Route headers on requests (other than the scenario's first INVITE) and Record-Route headers on responses inside Make Call, Receive Call, End Call, or other procedures provides support for testing Proxy servers that choose to stay in the message path between call participants.
- Whenever a procedure uses SIP Route for requests or Record-Route headers for responses, the Route or RecordRoute suffixes are appended to the procedure name.
- All call flows were created by using a SIP Allow header that advertises the INVITE, ACK, BYE, OPTIONS, CANCEL, SUBSCRIBE, NOTIFY, REFER, and MESSAGE methods.
- For most request messages, the SIP User-Agent header is present and the User-Agent value is defined in the SIP REGISTER message by using the "IxLoad-client"+\$UnitCh+"/v5.10" expression.
- All call flows were created so as to handle a large number of response messages from the 1xx, 2xx, 4xx, 5xx, 6xx classes, or the lack of response from test devices.

Test Objective Settings

- Configurations are provided for capacity and performance testing, the configured test objectives being Channels, Active Callers (AC), or Loops Initiated per Second (LPS) respectively. Generally, tests configured by using an LPS objective type have a number of channels specified in the **Custom Parameters** tab of the **Timeline and Objectives** page. The resulting TalkTime call parameter is used to configure one occurrence of media functions (Talk, Listen, or Voice Session) executed by the script scenario. For additional occurrences of media functions, the play time is specified in other modes than by using the TalkTime parameter. The Estimates Overhead parameter configured in the **Custom Parameters** tab takes into account delays introduced by the additional media functions, as well as Sleep functions or other functions that generate delays, used in a particular scenario.

Objective Settings	
Loops per Second Objective:	100
<input type="radio"/> Specify Talk Time	14000 ms
<input checked="" type="radio"/> Specify Number Of Channels	2000
Estimated Overhead Times:	5000 ms
Inter-loop Duration:	1000 ms
Active Scenario Channel:	0
<hr/>	
Calculated Talk Time:	14000

- Generally, tests are configured by using a RampUp time of two minutes (RampUp value of 25 users/s), a SustainTime of 1 hour, and a RampDown time of five minutes (RampDown value of 50 users/s).

Test Categories

The supplied test configurations fall into following categories:

- [Registration](#)
- [Basic Calls](#)
- [Advanced SIP Features](#)

For each test, the test configuration is described and information is provided on how to start from the existing configuration and adapt the test to your own needs.

Customizable Test Parameters

Although test configurations contained in the SIP Extended Functionality test suite are already configured, the following test parameters can be easily modified to adapt a test to your own testbed:

- IP address: The IP addresses are configured in a VoIPSIP activity's underlying Network page. Initially, test configurations use IP addresses starting with 20.1.1.1/16 for the first VoIPSIP activity, with 20.1.50.1/16 for the second VoIPSIP activity, and so on.

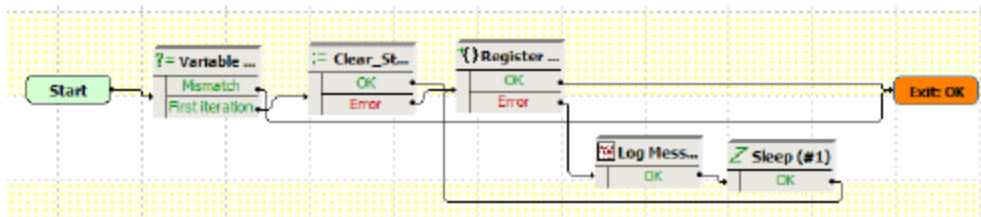
- Phone numbers: The source phone numbers are configured in a VoIPSIP activity's Dial Plan page. Initially, test configurations use the 160[00000000-] and 170[00000000-] sequence generating expressions for source and destination phone numbers.
- Proxy server IP address: For tests that use an external server, its IP address is configured in the VoIPSIP activity's SIP page.
- Domain name: Emulated UAs have the ixload-test domain configured in the VoIPSIP activity's SIP page.
- Chassis port: VoIPSIP activities are configured in the **Port Assignments** page by using a single port of an Acceleron load module board.

Predefined Common Test Procedures

Tests use some common registration and call procedures—Register and Make call—described in this section.

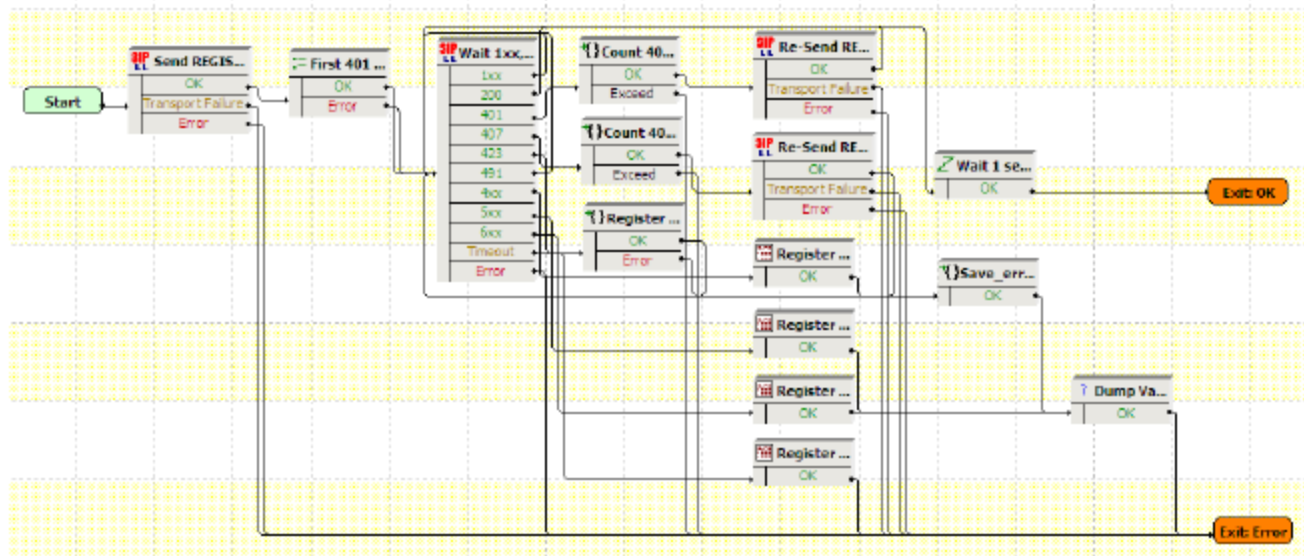
Register

The Register procedure performs registration on the first loop.



The initial Variable Test function tests if the current iteration is the first one, so as to execute registration only once, at the beginning of the test. The following Clear Statistics function resets the reg_sent_err and reg_rcv_err variables to an empty ("") value.

The actual registration is done by the Register procedure illustrated in the following image:



A SIP REGISTER message is sent by using the Send Register function, and then the count variable (which represents the occurrence of received 401 response message) is initialized to the '1' value.

The following Wait... function processes a large number of responses:

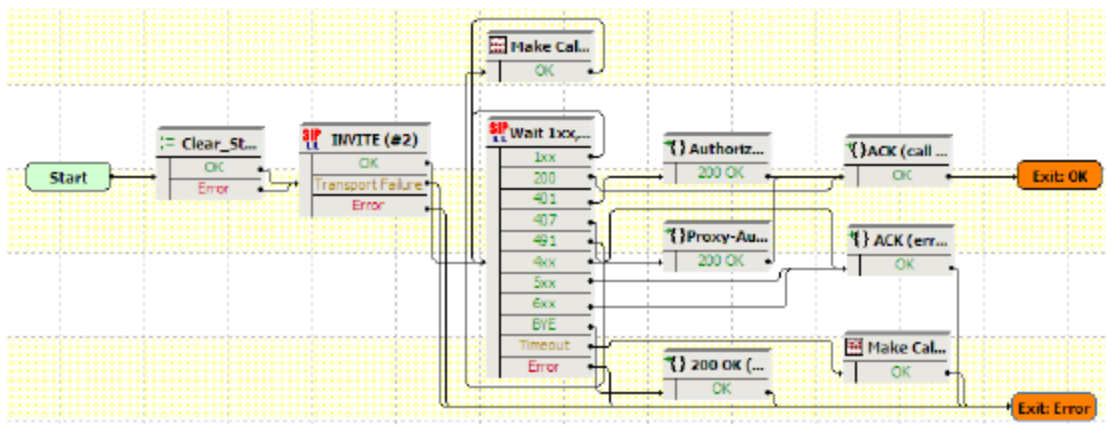
- For 1xx and 491 messages, no action is taken and the Wait... function loops.
- For 200 Ok messages, a Sleep function is executed and the Register procedure exits.

- For 401 messages, a Count 401 procedure verifies if the received 401 message is the first one; if this is the case, the REGISTER request is re-sent with authentication information and the script execution jumps to the Wait... function. Alternatively, in case this is a second 401 message received, which represents a permanent registration failure, the script execution starts anew from the initial Send Register function.
- For 407 messages, a Count 407 procedure verifies if the received 407 message is the first one; if this is the case, the REGISTER request is re-sent with authentication information and the script execution jumps to the Wait... function.
- For 423 messages, first an Extract Variable function is used for extracting the new_exp variable, then the REGISTER request is re-sent with an Expires header configured to this value.
- For 4xx messages other than 401, 407, 423, and 491, a message is logged by using the Log Message function, then the reg_sent_err and reg_rcv_err variables (containing the sent and received error messages) are extracted by using two Extract Variable functions inside the Save_error_messages procedure. The extracted strings are eventually written to logs by using the Dump Variable function.
- For 5xx and 6xx error messages, Log Message functions are triggered that log specific messages to the chassis port and that appear in the IxLoad **Event Viewer** window.

When the Register procedure encounters an error and exits on its Error output, a message is logged and the registration operation resumes, following an idle period enforced by a Sleep function.

Make Call

The Make Call procedure establishes a call.



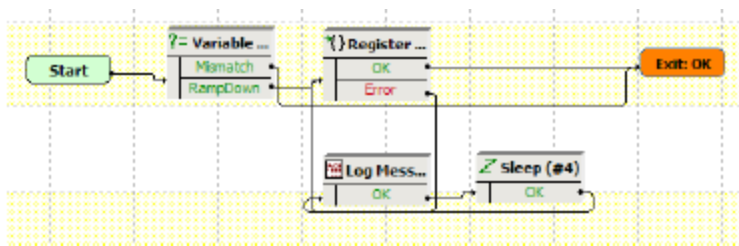
The procedure contains an initial Clear_Variables function that resets the proxy_auth string variable to an empty value. The call is initiated by using an Invite function, followed by a Wait function that processes a large number of response messages:

- For 1xx messages, no action is taken and the Wait... function loops.
- For 200 Ok messages, an ACK (call connected) procedure is executed that signals a successful call establishment.
- For 401 messages, an Authorization procedure is executed that first sends an ACK message and then re-sends the INVITE message with authentication information.

- For 407 messages, a Proxy-Authorization procedure is executed that sends an ACK message and then re-sends the INVITE message with authentication information.
- For 491 messages, a message is logged using the Log Message function and then execution resumes starting from the Wait... function.
- For 4xx (other than the above), 5xx, and 6xx error messages, an ACK (error) procedure is triggered that logs an error message and then extracts the call_sent_err and call_rec_err messages using Extract Variable functions. Finally, these error messages are dumped to the port (and to the Event Viewer pane) by using a Dump Variables script function.
- For BYE messages, a 200 Ok procedure is executed, and then the procedure exits on the Error output.

UnRegister

The UnRegister procedure performs de-registration while the test execution is on the ramp-down portion.



The procedure uses a Variable Test function to test if the execution is on the ramp-down portion (\$rampdown variable is equal to 1), and then executes the de-registration of SIP UAs.

Test Descriptions

For each test, the test configuration is described and information is provided on how to start from the existing configuration and adapt the test to your own needs.

Registration

The script in this category illustrates the registration of IxLoad-emulated SIP UAs with a Registrar server.

Register

This test illustrates the case of SIP UAs that register with a Registrar server.

Basic Calls

The scripts in this category illustrate the case of IxLoad-emulated SIP UAs that perform basic calls. Prior to initiating a call, UAs perform registration with a real Registrar by using the Register Complete procedure.

Basic_Call_Complete_Audio_LPS

This test illustrates UAs that establish calls to other UAs. After call establishment, endpoints exchange media using the Voice Session script function.

At activity level, no special settings are configured in addition to those described in [General Test Features](#).

At scenario level, registration is performed by using the Register procedure (see [Register](#)), while re-registration is configured by using the Enable session timers settings of the Automatic page. Call setup is done by using the Make Call procedure (see [Make Call](#)). Following the media exchange, endpoints finally de-register by using an Unregister procedure (see [UnRegister](#)).

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 200 Loops Initiated per Second (LPS).

Basic_Call_Complete_Audio_MCH

This test is similar with the [Basic_Call_Complete_Audio_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

Basic_Call_Complete_Multimedia_AC

This test is similar to [Basic_Call_Complete_Audio_LPS](#), with the difference that endpoints involved in the call exchange both audio and video media. The configured objective is 100 Active Callers.

Basic_Call_Complete_Multimedia_MCH

This test is similar with the [Basic_Call_Complete_Multimedia_AC](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

Basic_Call_Busy_LPS

This test illustrates SIP UAs that attempt to establish calls to other UAs, with the result that the connection is not established and a Busy 486 message is received instead.

At activity level, Caller and Callee register each with a Registrar server specified by an IP address and use an outbound Proxy server (SIP configuration page). Both parties have audio capability enabled (Audio configuration page).

At scenario level, the Caller attempts to establish a call with Callee by using a Make Call procedure (see [Make Call](#)), while the Callee responds by sending a Busy 486 message within a Receive Call-Busy Here procedure.

The configured test objective is 100 Loops Initiated per Second (LPS).

Basic_Call_Busy_MCH

This test is similar with the [Basic_Call_Busy_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

Basic_Call_Complete_Cancel_LPS

This test illustrates the case of SIP UAs that attempt to establish calls to other UAs, whereby the remote party does not answer the incoming call; the Caller eventually cancels the call request.

At activity level, Caller and Callee register each with a Registrar server specified by an IP address and use an outbound Proxy server (SIP configuration page). Both parties have the audio capability enabled (Audio configuration page).

At scenario level, the Caller attempts to establish a call with Callee by using a Make Call procedure. Because the remote party does not answer the incoming call, Caller eventually sends an SIP CANCEL message (by using a CANCEL Call procedure) to cancel the request.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved using a number of 2000 channels configured in the **Custom Parameters** tab.

Basic_Call_Complete_Cancel_MCH

This test is similar with the [Basic_Call_Complete_Cancel_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

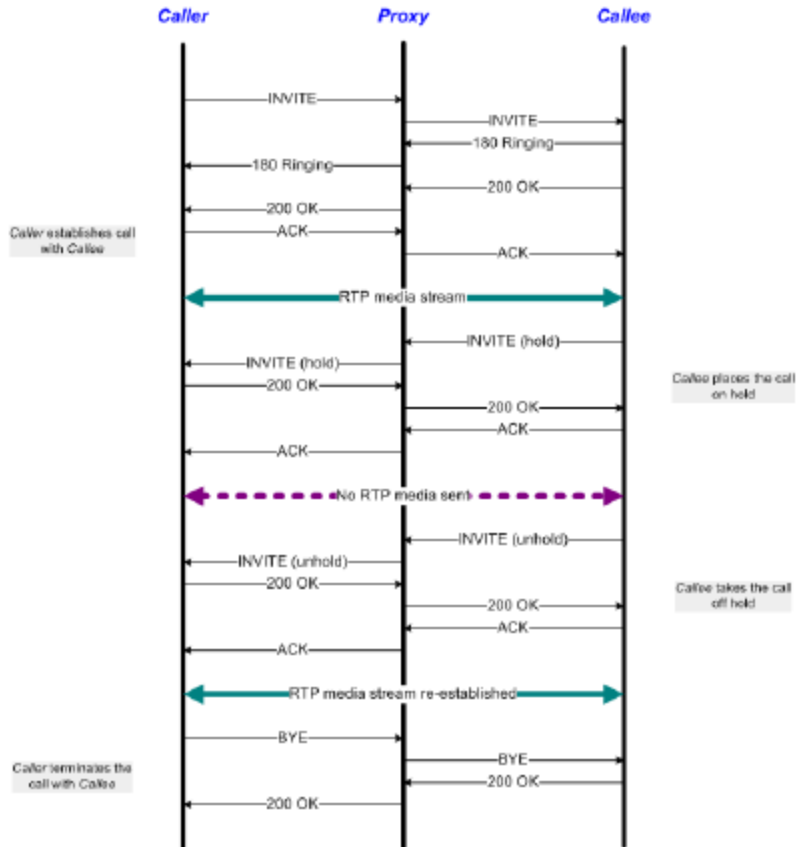
The configured test objective is 4000 Channels.

Advanced SIP Features

The scripts in this category illustrate the case of IxLoad-emulated SIP UAs that perform calls with other emulated UAs, whereby advanced SIP features, such as Call Hold, Call Transfer, Call Park, or 3-Way Calls, are used after call establishment.

Call_Hold_LPS

This test implements the call flow shown in the following image, whereby the Caller and the Callee are emulated by Ixload VoIPSIP activities, while the Proxy is a real device (DUT):



At activity level, Caller and Callee register each with a Registrar server specified by an IP address and use an outbound Proxy server (SIP configuration page). Both parties have the audio capability enabled (Audio configuration page).

At test scenario level, UAs establish calls to other UAs by using the Make Call procedure (see [Make Call](#)), and then exchange audio media over the call by using the Voice Session function. The remote party then puts the call on hold (SIP Hold-Initiate procedure) for a duration configured by the Sleep function and eventually takes the call off the hold state (SIP Unhold-Initiate procedure). Finally, endpoints exchange media again over the re-established call by using the Voice Session function.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved by using a number of 3000 channels configured in the **Custom Parameters** tab.

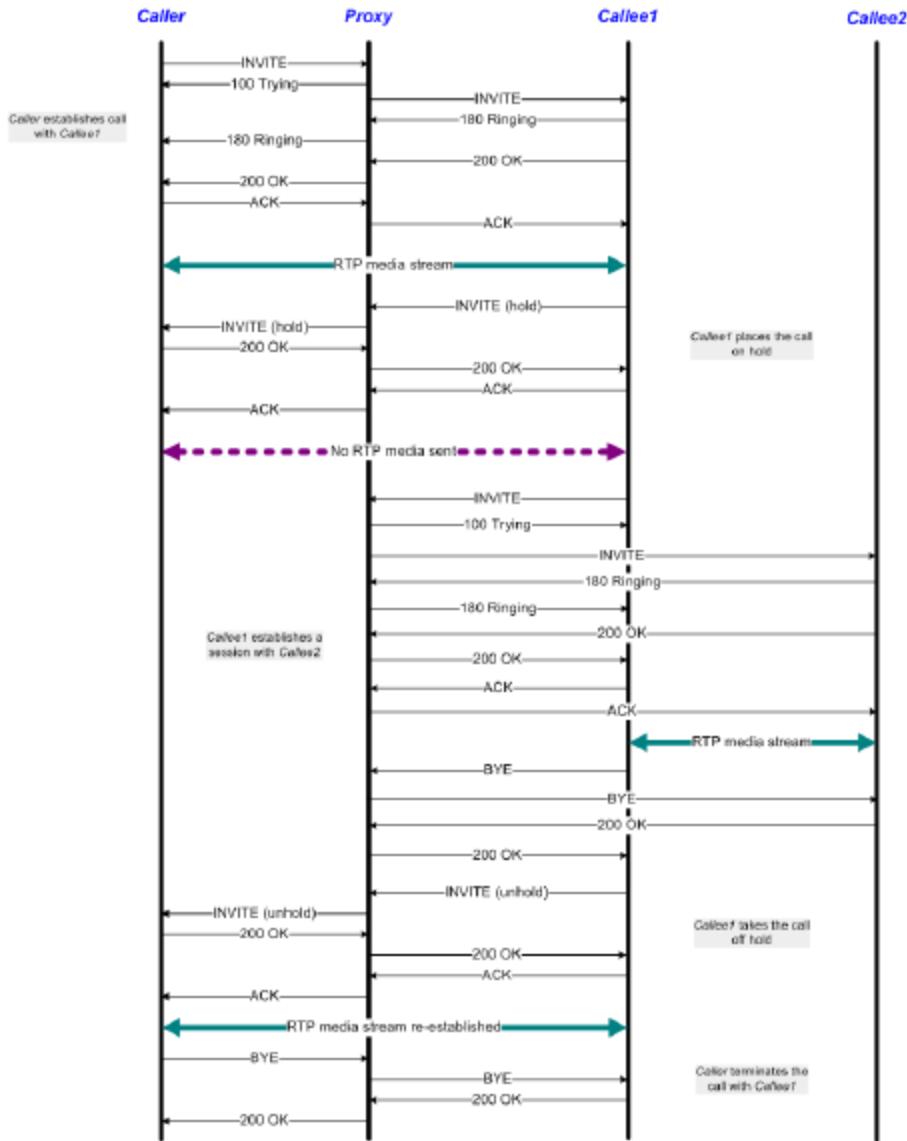
Call_Hold_MCH

This test is similar with the [Call_Hold_LPS](#) test, except for fact that multiple SIP UAs are configured to share the same IP address, that is, the IP address for SIP UAs parameter in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 4000 Channels.

Consultation_Hold_LPS

This test implements the call flow shown in the following image, whereby the Caller, Callee1, and Callee2 are emulated by Ixload VoIPSIP activities, while the Proxy is a real device (DUT):



At activity level, Caller, Callee1, and Callee2 register each with a Registrar server specified by an IP address and use an outbound Proxy server (SIP configuration page). All three parties have audio capability enabled (Audio configuration page).

At test scenario level, UAs establish calls to other UAs using the Make Call procedure, and then exchange audio media over the call (Voice Session function). Callee1 then puts the call with Caller on hold (SIP Hold-Initiate procedure) and establishes a call to another destination (Callee2) for a period of time configured by using the Sleep function. The call between destination (1) and (2) is terminated by Callee1 by using the SIP EndCall Initiate - Route procedure (BYE message contains a Route header), after which Callee takes the initial call off hold and exchanges media with Caller again by using the Voice Session function.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved by using a number of 3000 channels configured in the **Custom Parameters** tab.

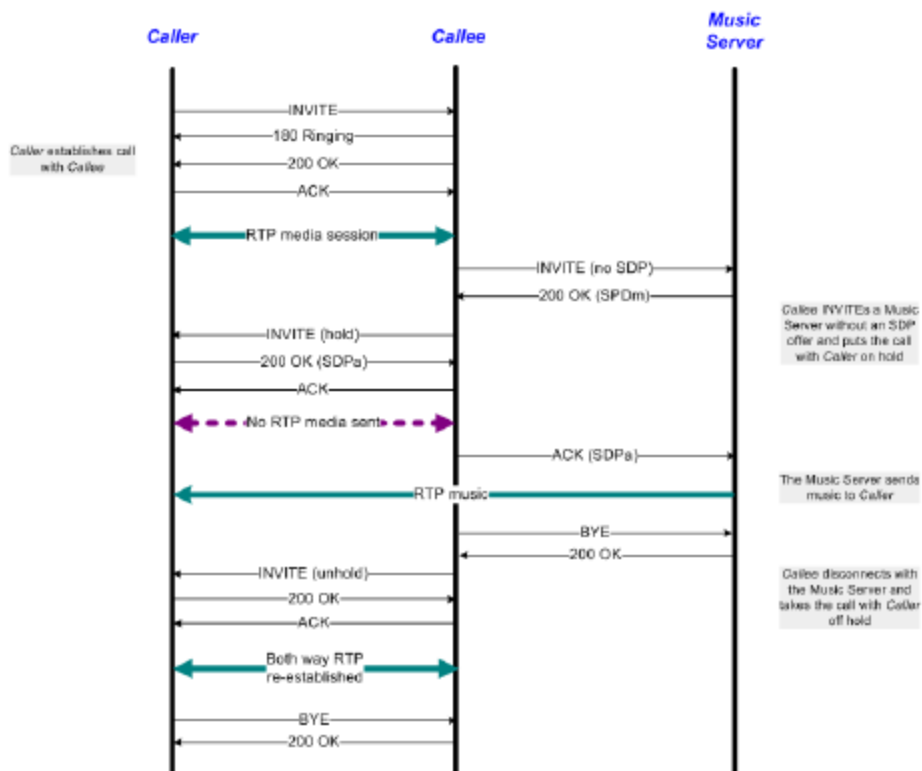
Consultation_Hold_MCH

This test is similar with the [Consultation_Hold_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Music_on_Hold_LPS

This test implements the call flow shown in the following image, whereby the Caller, the Callee, and the Music Server are emulated by Ixload VoIPSIP activities:



At activity level, Caller and Callee1 register each with a Registrar server specified by an IP address and use an outbound Proxy server (SIP configuration page). All three parties have audio capability enabled (Audio configuration page). As a note, the Music server is configured by using the **Use same value (per port)** setting for both SIP and RTP IP address allocation (Execution page).

At test scenario level, Caller establishes a call to Callee by using the Make Call procedure and exchanges audio media over the call (Voice Session function). The Callee then sends an INVITE to the Music Server without an SDP definition and puts the call with Caller on hold (SIP Hold-Initiate procedure). The Music server streams media to Caller (by using the Talk function), before Caller finally terminates the call by using a SIP EndCall Initiate procedure.

Eventually, the call between Callee and Caller is resumed by Callee who sends a re-INVITE to Caller by using a SIP Unhold - Initiate procedure.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved by using a number of 2000 channels configured in the **Custom Parameters** tab.

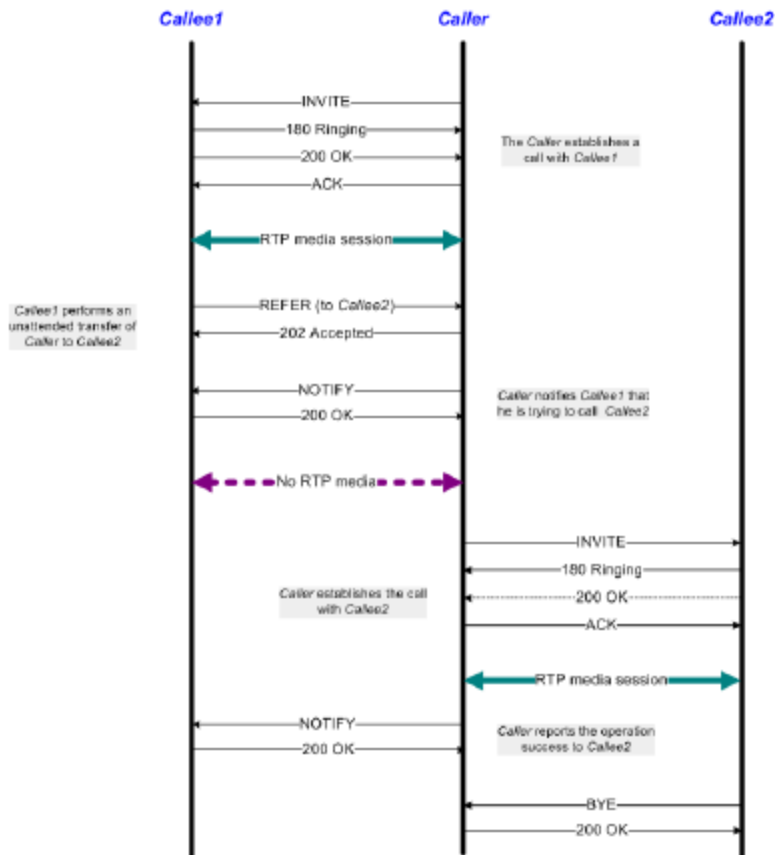
Music_on_Hold_MCH

This test is similar with the [Music_on_Hold_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Transfer_Unattended_LPS

This test implements the call flow shown in the following image, whereby the Caller, Callee1, and Callee2 are emulated by Ixload VoIPSIP activities:



At activity level Caller, Callee1, and Callee2 register each with a Registrar server specified by an IP address and use an outbound Proxy server (SIP configuration page). All three parties have audio capability enabled in the Audio configuration page.

At test scenario level, Caller establishes a call to Callee1 by using the Make Call procedure, and then exchange audio media over the call (Voice Session function). Callee1 then REFERS the Caller to Callee2 (Initiate Transfer procedure), with whom the Caller establishes a call and exchanges audio media. Finally, the Caller uses the Confirm Transfer procedure to notify the transfer completion to Callee1. The call is terminated by Callee2 by using an SIP EndCall Initiate procedure.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 50 Loops Initiated per Second (LPS), which is to be achieved by using a number of 2000 channels configured in the **Custom Parameters** tab.

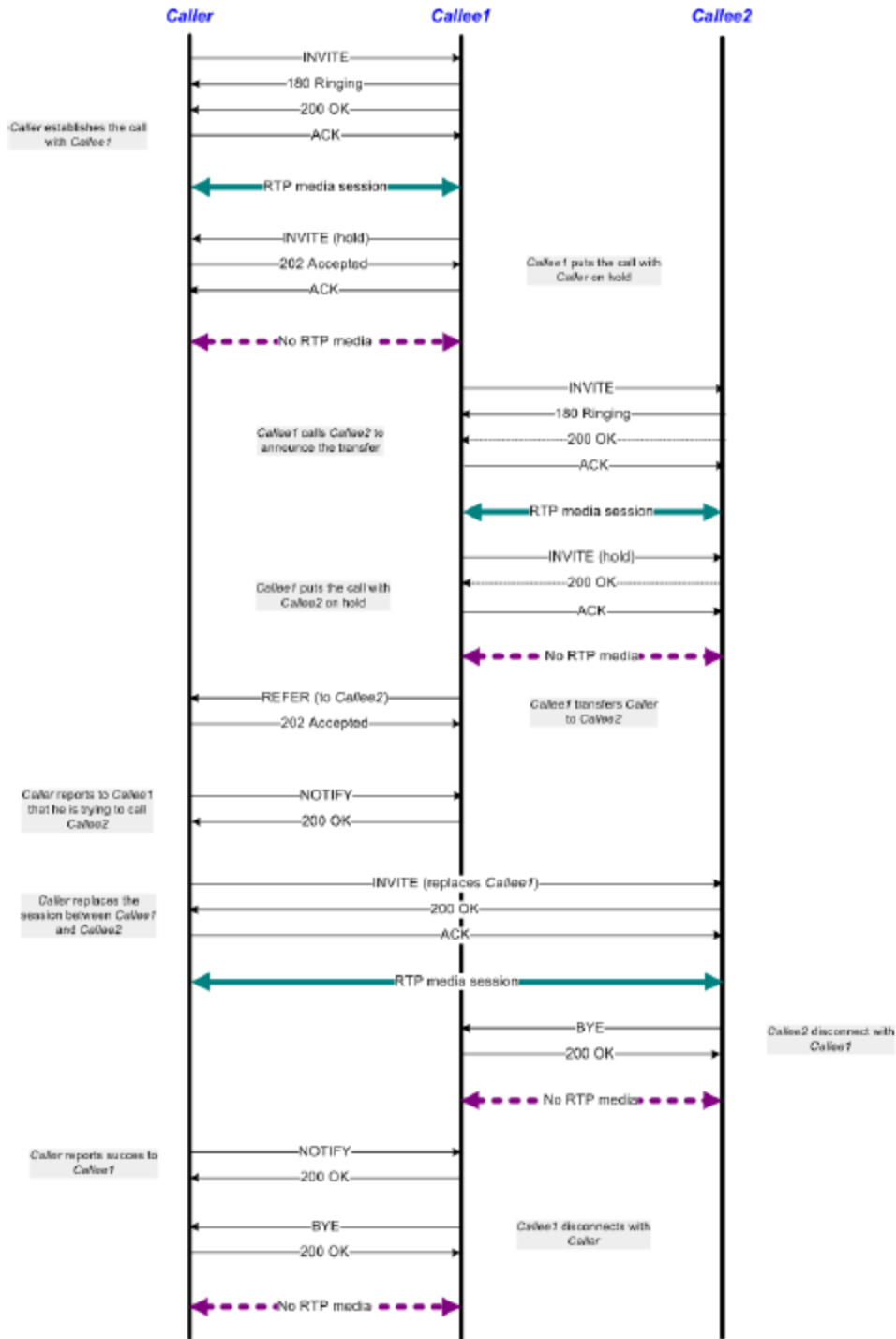
Transfer_Unattended_MCH

This test is similar with the [Transfer_Unattended_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity’s Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Transfer_Attended_LPS

This test implements the call flow shown in the following image, whereby the Caller, Callee1, and Callee2 are emulated by Ixload VoIPSIP activities. The test is similar to the previous one, except that a consultation with Callee2 occurs before transferring the call.



At activity level, Caller, Callee1 and Callee2 register each with a Registrar server specified by an IP address and use an outbound Proxy server (SIP configuration page). All three parties have audio capability enabled in the Audio configuration page.

At test scenario level, Caller establishes a call with Callee1 by using the Make Call procedure, and then exchange audio media over the call (Voice Session function). Callee1 then puts the call with Caller on hold (SIP Hold-Initiate procedure) and establishes a consultation session with Callee2. After performing media exchange, Callee1 puts the call with Callee2 on hold and REFERS Caller to Callee2 (Initiate Transfer procedure). Caller replaces Callee1 (Make Call procedure containing an INVITE message with a Replaces header) and establishes a session with Callee2. This call is terminated by Callee2 using a SIP EndCall Initiate procedure.

Finally, Callee1 terminates the Call with Caller by using a similar SIP EndCall Initiate procedure.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 50 Loops Initiated per Second (LPS), which is to be achieved using a number of 1500 channels configured in the **Custom Parameters** tab.

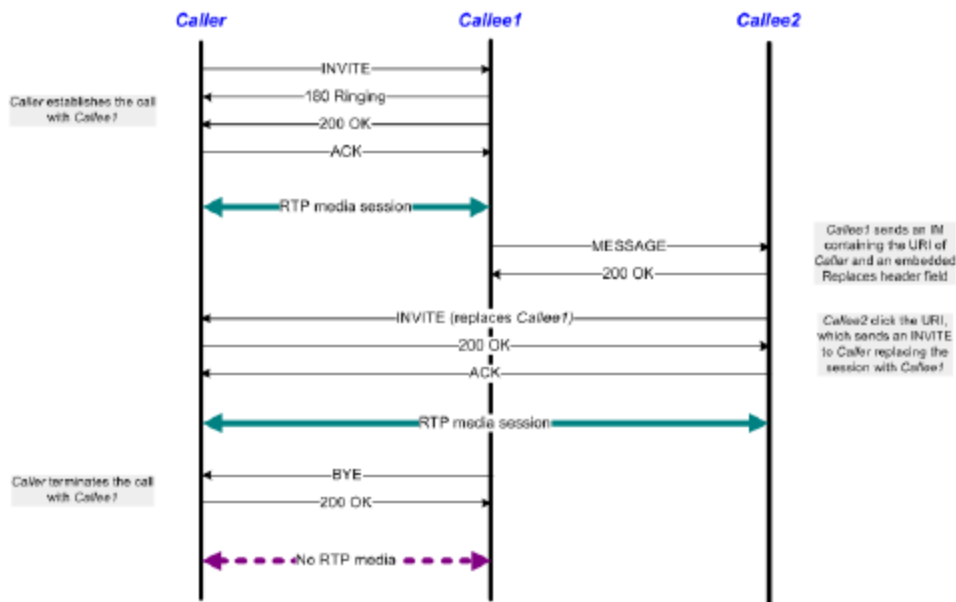
Transfer_Attended_MCH

This test is similar with the [Transfer Attended LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Transfer_Instant_Messaging_LPS

This test implements the call flow shown in the following image, whereby the Caller, Callee1, and Callee2 are emulated by Ixload VoIPSIP activities:



At activity level, Caller, Callee1 and Callee2 register each with a Registrar server specified by an IP address and use an outbound Proxy server (SIP configuration page). All three parties have audio capability enabled (Audio configuration page).

At test scenario level, Caller establishes a call with Callee1 by using the Make Call procedure and the parties exchange audio media over the call by using the Voice Session function. Dialog information is then provided by Callee1 to Callee2 by using a Send Instant Message procedure (containing a MESSAGE command). Callee2 sends Caller an INVITE message (by using a Make Call procedure) that contains a Replaces header and the two parties establish a session with audio media exchange. The call is eventually terminated by Callee2 by using a SIP End-Call Initiate procedure.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved using a number of 2000 channels configured in the **Custom Parameters** tab.

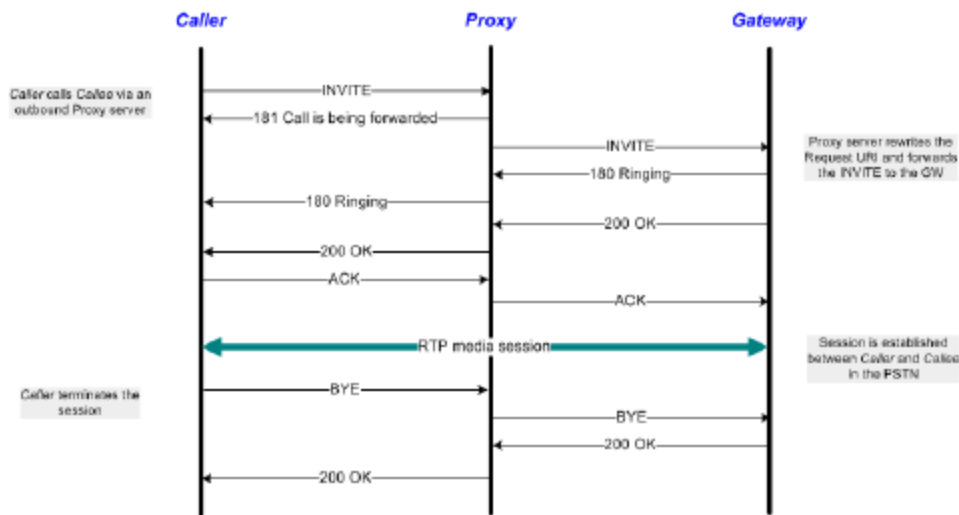
Transfer_Instant_Messaging_MCH

This test is similar with the [Transfer_Instant_Messaging_LPS](#), except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Call_Forwarding_Unconditional_LPS

This test implements the call flow shown in the following image, whereby the Caller and the Gateway are emulated by Ixload VoIPSIP activities, while the Proxy is a real device (DUT):



At activity level, the Caller registers with the Registrar server specified by its IP address and uses an outbound Proxy server (SIP configuration page). Both the Caller and the Gateway have audio capability enabled (Audio configuration page).

At scenario level, the Caller tries to place a call to a PSTN number, while the Proxy server routes the call to a Gateway. On the established call, media is exchanged by using the Voice Session function. Eventually, the Caller terminates the call sending a BYE message (within the SIP EndCall Initiate procedure) that contains a Route header.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved by using a number of 2000 channels configured in the **Custom Parameters** tab.

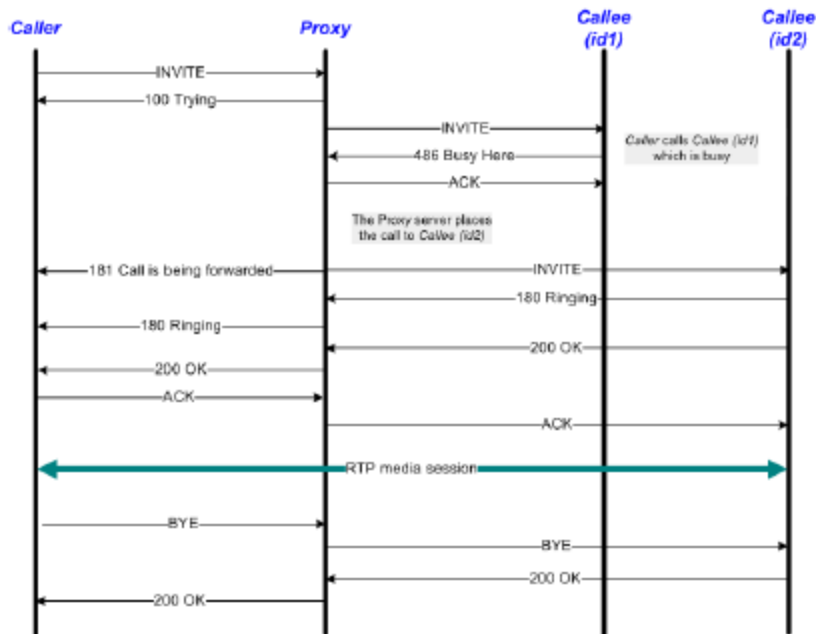
Call_Forwarding_Unconditional_MCH

This test is similar with the [Call_Forwarding_Unconditional_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Call_Forwarding_on_Busy_LPS

This test implements the call flow shown in the following image, whereby the Caller, Callee(id1), and Callee(id2) are emulated by Ixload VoIPSIP activities, while the Proxy is a real device (DUT):



At activity level, Caller and Callee register each with a Registrar server specified by its IP address (SIP configuration page). Both Callee identities have the same phone number configured (**Dial Plan** page), and both the Caller and the Callee have audio capability enabled (Audio configuration page).

At scenario level, the Caller attempts to place a call to Callee, with the Proxy routing the call first to Callee (id1), who replies with a '486 Busy here' response, and then to Callee (id2). The call is accepted by Callee (id2) by using a SIP Receive Call procedure that contains '180 Ringing' and '200 OK' messages. After exchanging media over the established call by using the Voice Session function, the Caller terminates the call sending a BYE message containing a Route header (SIP EndCall Initiate - Route procedure).

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved by using a number of 2000 channels configured in the **Custom Parameters** tab.

Call_Forwarding_on_Busy_MCH

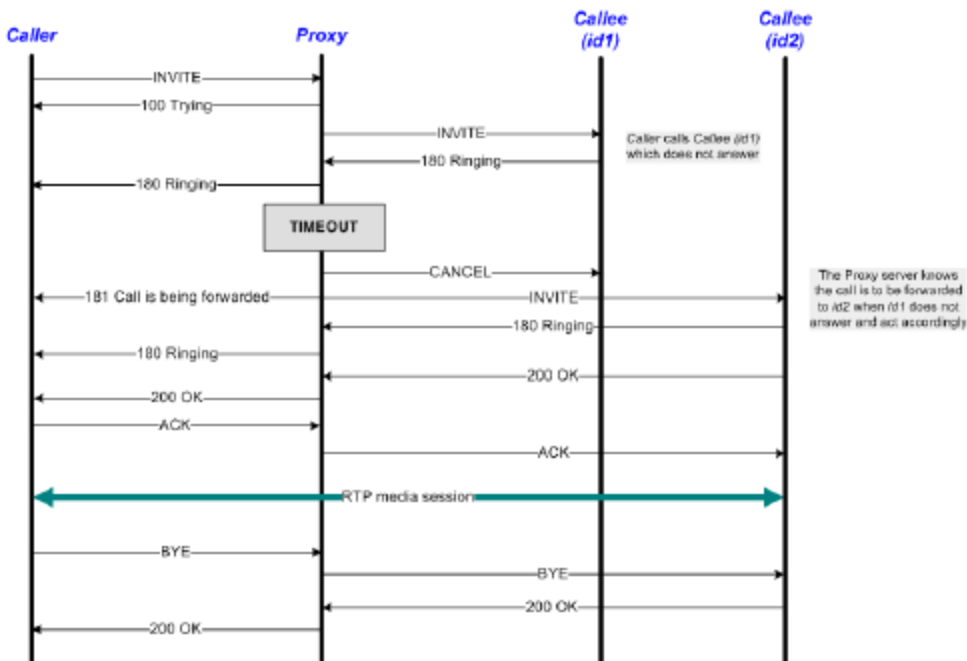
This test is similar with the [Call_Forwarding_on_Busy_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping the same value (per port) value.

The configured test objective is 8000 Channels.

Call_Forwarding_on_No_Answer_LPS

This test implements the call flow shown in the following image, whereby the Caller, Callee(id1) and Callee(id2) are emulated by Ixload VoIPSIP activities, while the Proxy is a real device (DUT).

The test is similar to the previous [Call_Forwarding_on_Busy_LPS](#) test, except that the call is forwarded by the Proxy server on a no answer timeout condition instead of a busy condition.



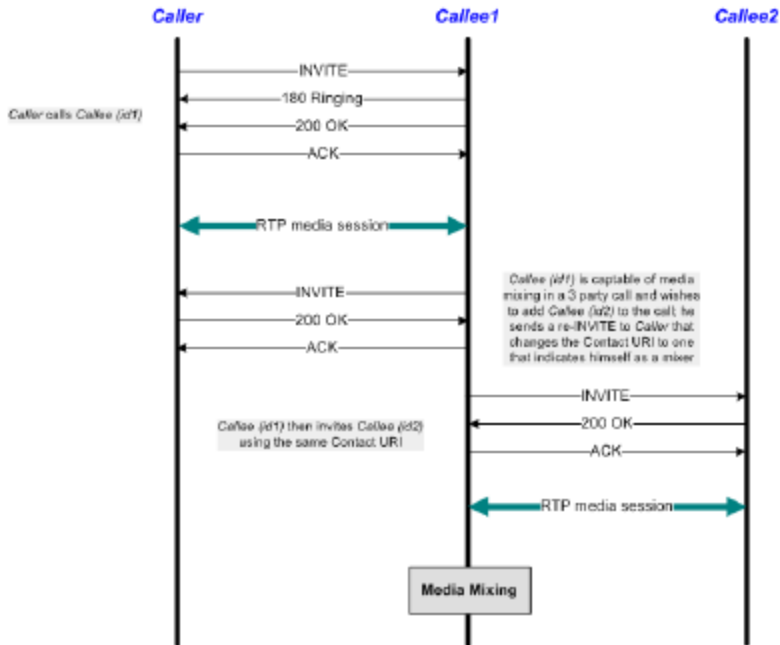
Call_Forwarding_on_No_Answer_MCH

This test is similar with the [Call_Forwarding_on_No_Answer_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Three_Way_Conference-Third_Party_Added_LPS

This test implements the call flow shown in the following image, whereby the Caller, Callee1, and Callee2 are emulated by Ixload VoIPSIP activities:



At activity level, Caller, Callee1, and Callee2 register each with a Registrar server specified by its IP address (SIP configuration page). All VoIPSIP activities have audio capability enabled (Audio configuration page).

At scenario level, the Caller places a call to Callee1 and the party exchange audio media. Callee1 then uses a Make Call procedure to re-invite Caller, and then uses another Make Call procedure for inviting Callee2.

After exchanging media over the established call by using the Voice Session function, the Caller terminates the call sending a BYE message within SIP EndCall Initiate - Route procedure.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved by using a number of 1000 channels configured in the **Custom Parameters** tab.

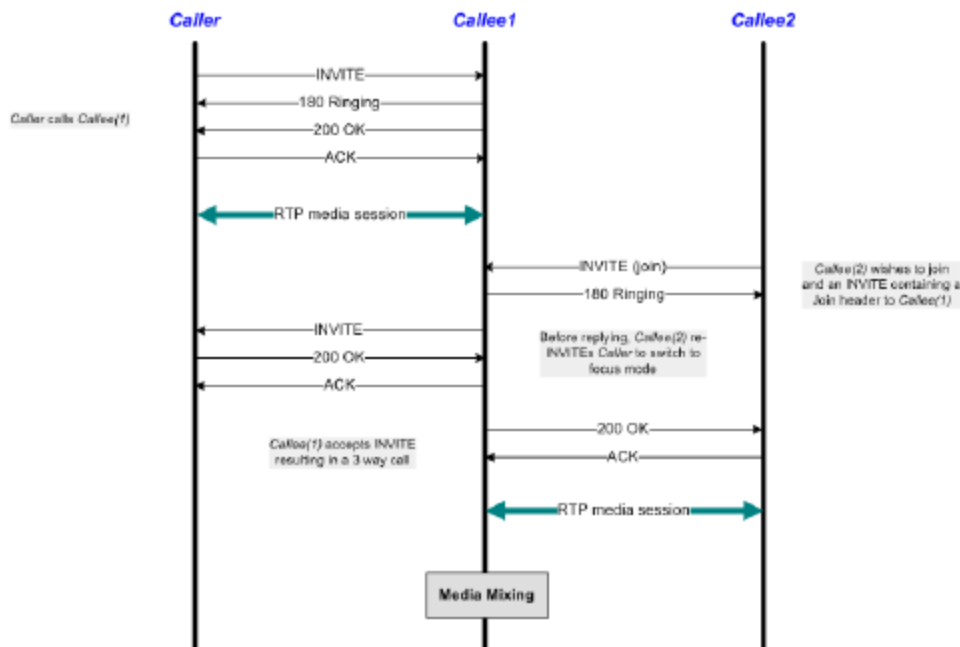
Three_Way_Conference-Third_Party_Added_MCH

This test is similar with the [Three_Way_Conference-Third_Party_Joins_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Three_Way_Conference-Third_Party_Joins_LPS

This test implements the call flow shown in the following image, whereby the Caller, Callee1, and Callee2 are emulated by Ixload VoIPSIP activities.



At activity level, Caller, Callee1, and Callee2 register each with a Registrar server specified by its IP address (SIP configuration page). All VoIPSIP activities have audio capability enabled (Audio configuration page).

At scenario level, the Caller places a call to Callee1 and the parties exchange audio media. Callee2 joins the conference by using a Make Call procedure containing an INVITE message with a Join header. Callee1 also re-INVITEs Caller to the focus mode using another Make Call procedure.

After exchanging media over the established call by using the Voice Session function, the Caller terminates the call sending a BYE message within a SIP EndCall Initiate - Route procedure.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved by using a number of 1000 channels configured in the **Custom Parameters** tab.

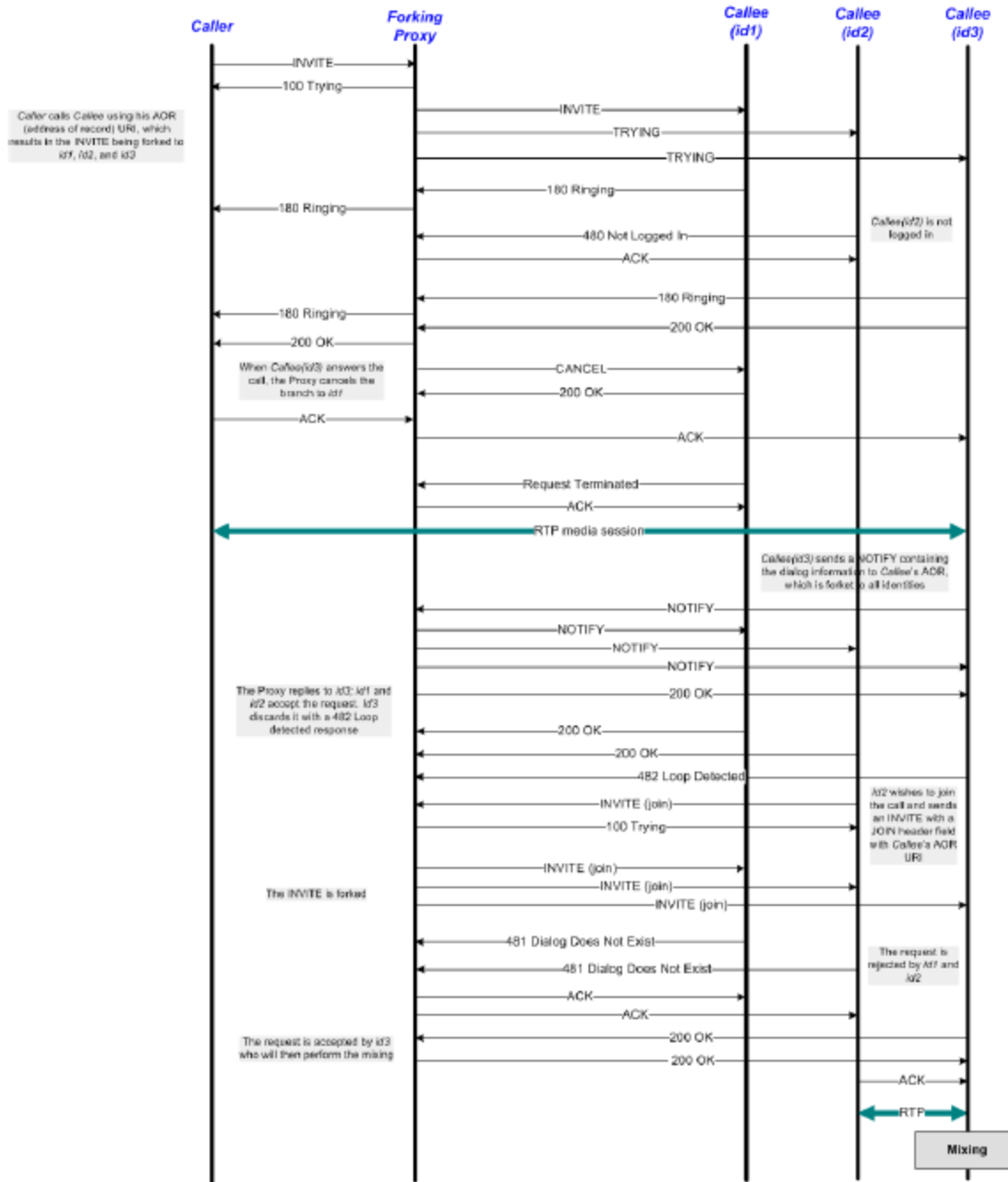
Three_Way_Conference-Third_Party_Joins_MCH

This test is similar with the [Three_Way_Conference-Third_Party_Joins_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Single_Line_Extension_LPS

This test implements the call flow shown in the following image, whereby the Caller, Callee(id1), Callee(id2), and Callee(id3) are emulated by Ixload VoIPSIP activities, and the Forking Proxy is a real device (DUT).



At activity level, all Callee identities have the same phone number configured.

At scenario level, Caller tries to place a call to Callee, while the Proxy routes the call in turn to all known identities (id1, id2, id3, and id4), until the call is established successfully. Callee(id3) sends a NOTIFY message containing the dialog information to Callee's Address-of-Record, and Callee(id2) attempts to join the call by using an INVITE message with a Join header. The request is accepted by Callee(id3) who perform the media mixing.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved by using a number of 1000 channels configured in the **Custom Parameters** tab.

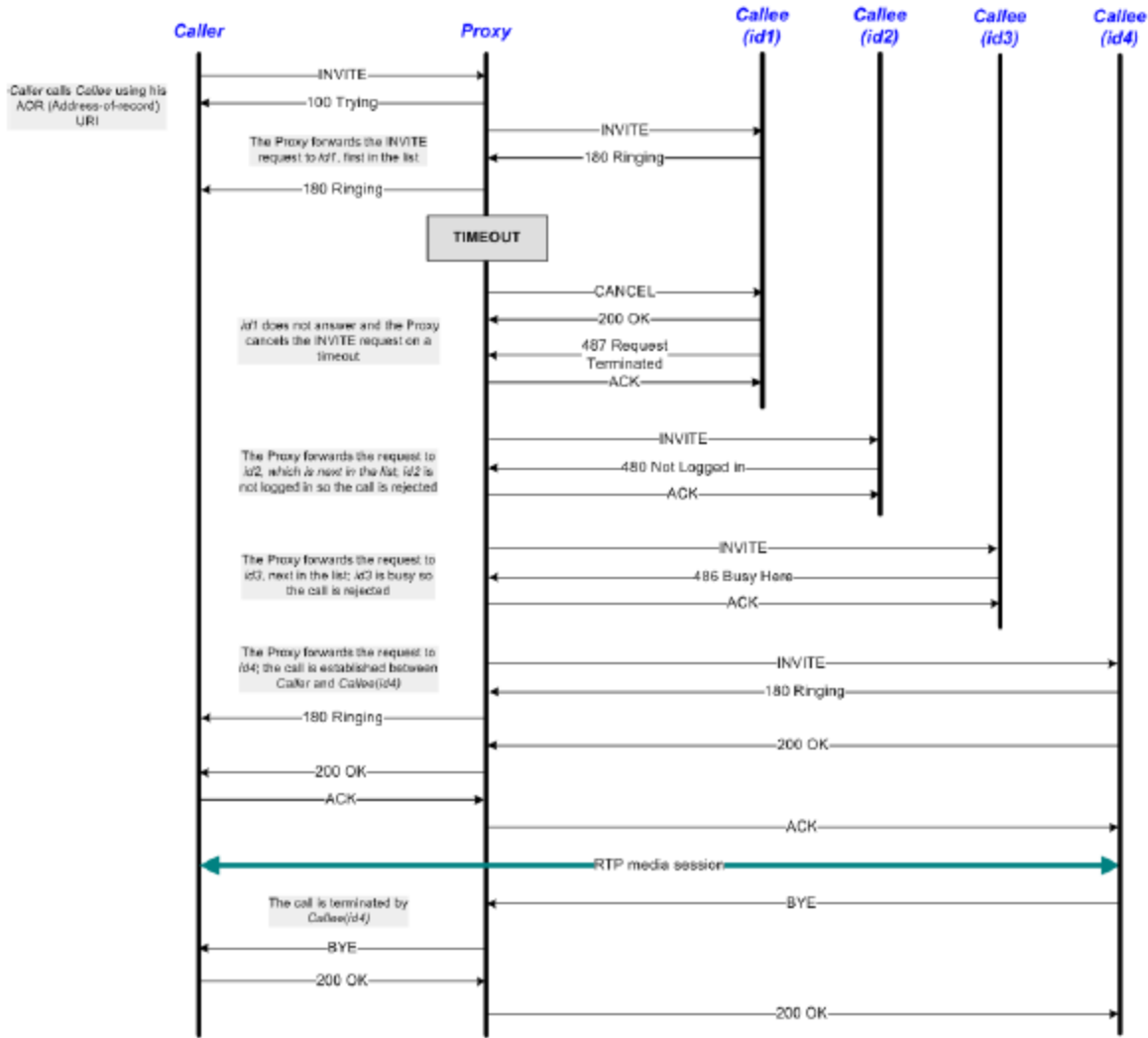
Single_Line_Extension_MCH

This test is similar with the [Single_Line_Extension_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Find-Me_LPS

This test implements the call flow shown in the following image, whereby the Caller, Callee(id1), Callee(id2), and Callee(id3) are emulated by Ixload VoIPSIP activities, while the Proxy server is a real device.



At activity level, all Callee identities are configured by using the same phone number.

At scenario level, Caller tries to place a call to Callee by using a Make Call procedure, while the Proxy routes the call in turn to all known Callee identities (id1, id2, id3, and id4), until the call is established successfully with Callee(id4). After exchanging media by using the Voice Session function, Callee(id4) terminates the call by sending a BYE message within a SIP EndCall Initiate - Route procedure.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved by using a number of 1000 channels configured in the **Custom Parameters** tab.

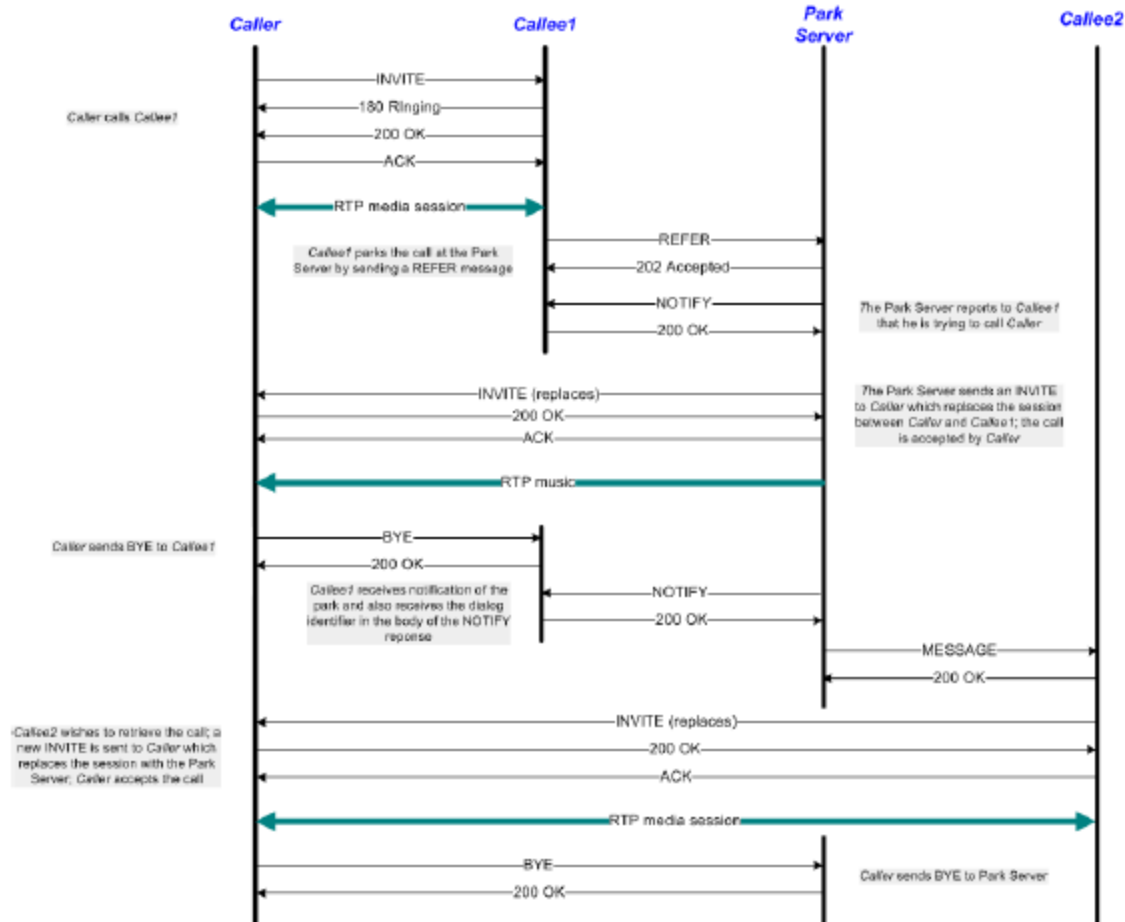
Find-Me_MCH

This test is similar with the [Find-Me_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Call_Park_LPS

This test implements the call flow shown in the following image, whereby the Caller, Callee1, and Callee2 are emulated by Ixload VoIPSIP activities, and the Park server is a real device (DUT).



At activity level, no special settings are configured other than those described in [General Test Features](#).

At scenario level, Caller and Callee1 establish a call with audio media exchange, and then Callee1 parks the call by sending a REFER (Replaces=<call-id>) message to the Park server. The Park server NOTIFYs Callee1 that they are trying to complete the operation and sends to the Caller an INVITE (Replaces:<call-id>) message, also containing an SDP definition. Caller accepts the request and music is streamed from the server over the established session using a Talk function.

Finally, Callee2 wishes to retrieve the call and sends an INVITE with a Replaces header to Caller (Make Call procedure). After being accepted by Caller, the Park server is replaced by Callee2 in the call.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 50 Loops Initiated per Second (LPS), which is to be achieved by using a number of 2000 channels configured in the **Custom Parameters** tab.

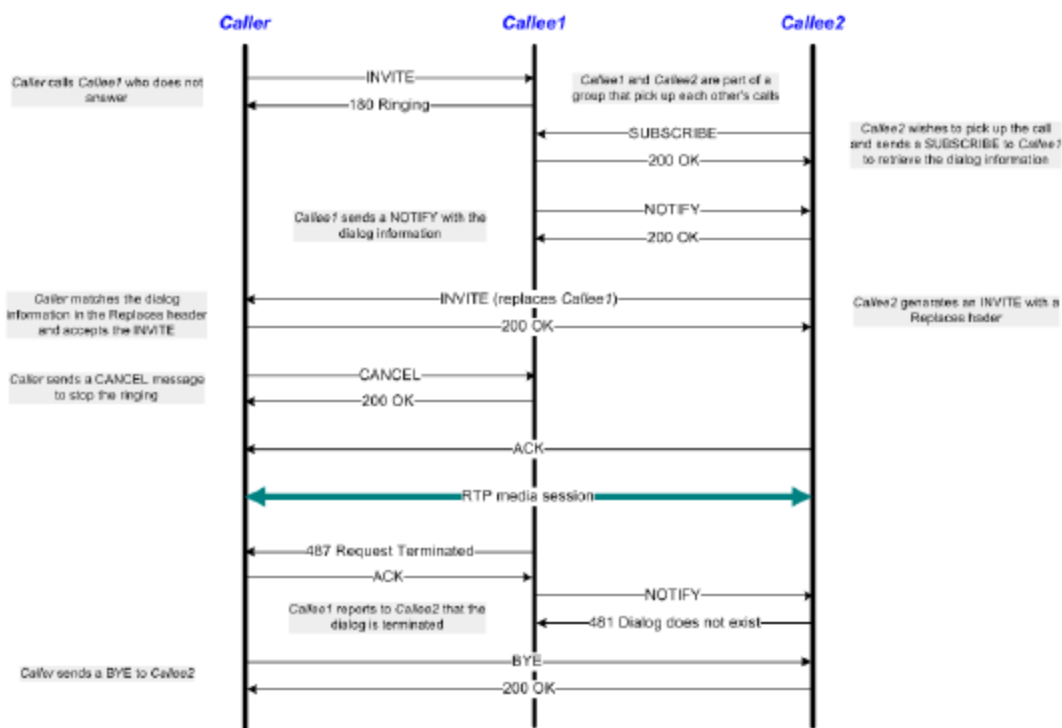
Call_Park_MCH

This test is similar with the [Call_Park_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Call_Pickup_LPS

This test implements the call flow shown in the following image, whereby the Caller, Callee1, and Callee2 are emulated by Ixload VoIPSIP activities.



At activity level, no special settings are configured other than those described in [General Test Features](#).

At scenario level, by using a Make Call procedure, Caller places a call to Callee1, who is configured in the same call group as Callee2. Callee2 sends a SUBSCRIBE (Event:dialog) message to be able to pick up calls addressed to Callee1. When Callee1 is then called, a NOTIFY message with the dialog information (dialog-info field) is sent to Callee2, who replaces Callee1 in the call with Caller.

The call between Caller and Callee2 is eventually terminated by Caller who sends a BYE message within a SIP EndCall Initiate - Route procedure.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved by using a number of 2000 channels configured in the **Custom Parameters** tab.

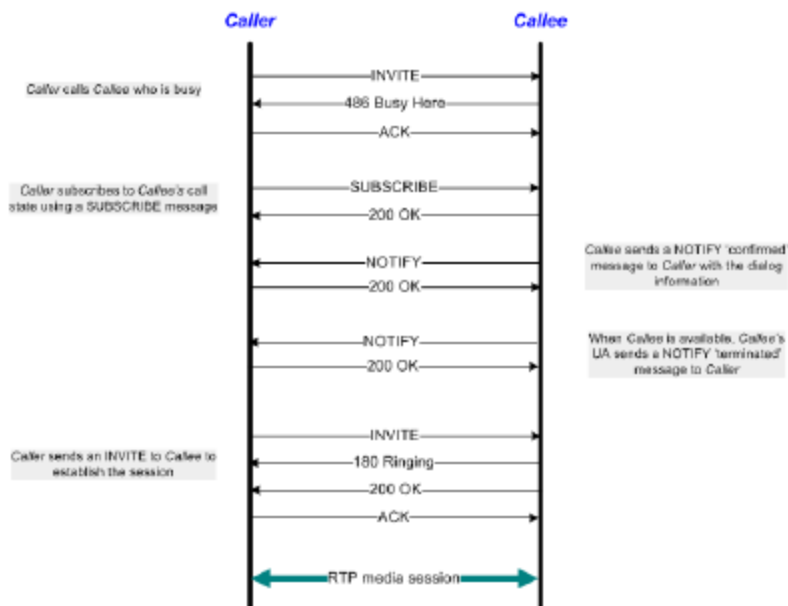
Call_Pickup_MCH

This test is similar with the [Call_Pickup_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Automatic_Redial_LPS

This test implements the call flow shown in the following image, whereby the Caller and the Callee are emulated by Ixload VoIPSIP activities.



At activity level, no special settings are configured other than those described in [General Test Features](#).

At scenario level, after having made an initial call that results in a busy condition, Caller sends a SUBSCRIBE (Event=dialog) message, and then waits for a NOTIFY message by using the Wait for Availability Information procedure. Following the receiving of the notification, Caller establishes the call by using a Make Call procedure. The established call is eventually terminated by Caller who sends a BYE message within a SIP EndCall Initiate - Route procedure.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved by using a number of 2000 channels configured in the **Custom Parameters** tab.

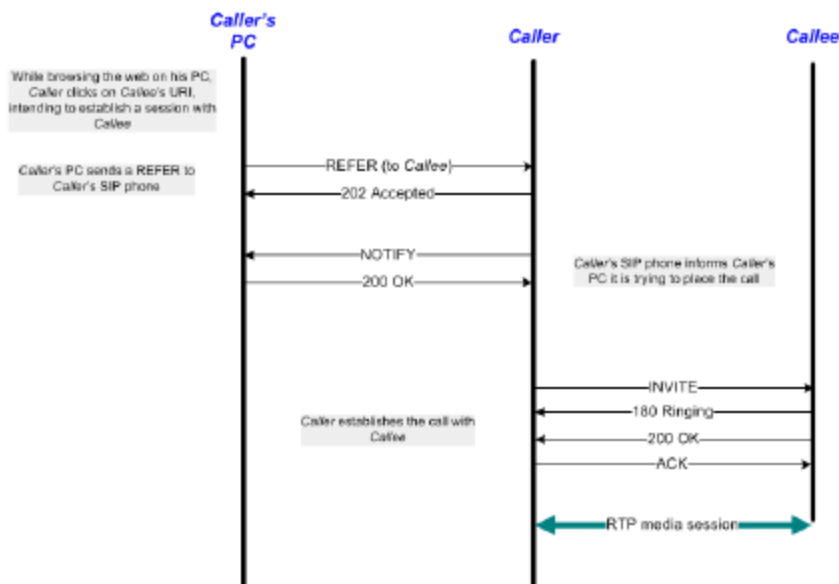
Automatic_Redial_MCH

This test is similar with the [Automatic_Redial_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

Click_to_Dial_LPS

This test implements the call flow shown in the following image, whereby the Caller, the Caller's PC, and the Callee are emulated by Ixload VoIPSIP activities.



At activity level, the Caller has no destination phone number configured, with this information being conveyed to him by a REFER message.

At scenario level, a Refer To procedure is used to refer the Callee to Caller. Caller then uses a Make Call procedure to establish the call and exchanges audio media by using a Voice Call function. The

established call is eventually terminated by Caller who sends a BYE message within a SIP EndCall Initiate - Route procedure.

The test scenario provides support for the SIP Route/Record-Route mechanism as described in [Scenario Settings](#).

The configured test objective is 100 Loops Initiated per Second (LPS), which is to be achieved by using a number of 1000 channels configured in the **Custom Parameters** tab.

Click_to_Dial_MCH

This test is similar with the [Click_to_Dial_LPS](#) test, except for the fact that SIP UAs are configured as sharing the same IP address for both signaling and media, that is, the Channel mapping rules for SIP UAs and Channel mapping rules for media parameters in the VoIPSIP activity's Execution page is set to **Use the same value (per port)** value.

The configured test objective is 8000 Channels.

This page intentionally left blank.

APPENDIX A Creating a SIP Message from Template

The **Create from Template** window helps you create correct SIP messages by providing templates for SIP message parameters.

When creating a new SIP message by using the **Create from Template** window, you can take either of the approaches below:

- Start from a generic message and select the headers to include in the SIP message.
- Load and subsequently modify a predefined message template.

[Table A-1](#), [Table A-2](#), and [Table A-3](#) describe the main parameters and options that you can set in the **Structured Message** window. For more details about these parameters, see the following:

- RFC 3261 – Session Initiation Protocol
- RFC 3265—Session Initiation Protocol—Specific Event Notification
- RFC 3515 – The Session Initiation Protocol Refer Method
- RFC 3262 – Reliability of Provisional Responses in SIP
- RFC 2976 – The SIP INFO Method
- RFC 3311 – The SIP Update Method
- RFC 3455 – Private Header (P-Header) Extensions to the Session Initiation Protocol (SIP) for the 3rd-Generation Partnership Project (3GPP)
- RFC 3325 – Private Extensions to the Session Initiation Protocol (SIP) for Asserted Identity within Trusted Networks
- draft - ietf - sip - replaces - 04
- draft - ietf - referred by -05
- draft - ietf - sipping - cc - transfer - 02

SIP Message Elements

Depending on the message type, SIP messages comprise the following elements:

- A Request line (request) or a Status Line (response)
- A variable number of message headers
- Message body

The **Create from Template** window enables you only to build the message first line and the message headers part, it has no influence on defining the message body.

While the message headers are common to both request and response messages, first line parameters are different, depending on whether the message is a SIP request or a SIP response.

[Table A-1](#), [Table A-2](#), and [Table A-3](#) list the specific request and response parameters, as well as the common parameters.

- Specific Request Parameters
- Specific Response Parameters
- Common Parameters

The **Parameters** page contains information about the structure and content of the message body, but to determine how the message looks like, you must visit the Behavior and Flow Manager pages.

Specific Request Parameters

SIP Request parameters are the request line and the message headers parameters.

When clicking **Create From Template** for a request message, the specific request parameters listed in [Table A-1](#) are available.

Option	Description
Current Template	The message template name that is currently loaded.

Load Template	<p>Opens the Load SIP Message Template dialog box and allows you to select the SIP message template to load. The available templates are as follows:</p> <p><Generic></p> <ul style="list-style-type: none"> • INVITE • ACK • OPTIONS • BYE • CANCEL • REGISTER • NOTIFY • SUBSCRIBE • REFER • MESSAGE • PRACK • INFO • UPDATE
Parameter (list)	<p>The list with available parameters and the appropriate settings. The Request-Line comprises the following main request parameters:</p> <ul style="list-style-type: none"> • Method • Request-URI • SIP-Version

Specific Response Parameters






Option	Description
--------	-------------






Current Template	<p>The SIP status template name that is currently loaded. The status templates are defined as combinations of SIP message templates and status codes.</p> <p>SIP messages templates:</p> <ul style="list-style-type: none"> • <Generic> • INVITE • ACK • OPTIONS • BYE • CANCEL • REGISTER • NOTIFY • SUBSCRIBE • REFER • MESSAGE • PRACK • INFO • UPDATE
Load Template	<p>Opens the Load SIP Template dialog box.</p> <p>The available message templates and response codes are mentioned in the preceding row.</p>
Parameter (list)	<p>The list of available parameters and the appropriate settings. The main Status line response parameters are as follows:</p> <ul style="list-style-type: none"> • SIP-Version • Status-Code • Reason-Phrase

Common Parameters

In addition to the first line of the message—a Request line for requests and a Status line for responses—SIP messages comprise message headers, which are common to both requests and responses. The following table (Table A-3) includes a description of the common parameters:

Option	Description
--------	-------------

Parameters (list)	<p>Available parameters and appropriate settings:</p> <p>Message Headers:</p> <p>Accept, Accept-Encoding, Accept-Language, Alert Info, Allow, Authentication-Info, Authorization, Call-ID, Call-Info, Contact, Content-Disposition, Content-Encoding, Content-Language, Content-Length, Content-Type, Cseq, Date, Error-Info, Event, Expires, From, In-Reply-To, Max-Forwards, Min-Expires, MIME Version, Organization, P-Associated-URI, P-Called-Party-ID, P-Visited-Network-ID, P-Access-Network-Info, PCharging-Function-Addresses, P-Charging-Vector, PAsserted-Identity, P-Preferred-Identity, Priority, Proxy-Authenticate, Proxy-Authorization, Proxy-Require, Record-Route, Referred-By, Refer-To, Replaces Reply-To, Require, Retry-After, Route, Server, Subject, Subscription-State, Supported, Timestamp, To, Unsupported, User-Agent, Via, Warning, WWW-Authenticate, extension-header.</p> <hr/> <p> Note: RSeq and Rack message headers are used only by PRACK and 1xx responses.</p> <hr/> <p>The following private headers (P-headers) define specific extensions to SIP required for IMS architecture (3GPP) testing:</p> <ul style="list-style-type: none"> • P-Associated-URI (RFC 3455) • P-Called-Party-ID (RFC 3455) • P-Visited-Network-ID (RFC 3455) • P-Access-Network-Info (RFC 3455) • P-Charging-Function-Addresses (RFC 3455) • P-Charging-Vector (RFC 3455) • P-Asserted-Identity (RFC 3325) • P-Preferred-Identity (RFC 3325) <p>By default, each message header has an index (for example, (#01)) included in its name string. This is because the user can duplicate an existing message header, in which case the duplicate has the number incremented (for example, (#02)).</p>
Color Coding	<p>The following color codes are used to show different headers:</p> 
 Duplicate	Duplicates the selected message header.
 Delete	Deletes the selected message header. This option is available only if the message has at least one copy.
 Move UP	Moves the selected message header up.

 Move Down	Moves the selected message header down.
<div style="border: 1px solid black; padding: 5px;">  Note: Moving Up/Down the headers establishes the order in which they are matched by the Wait functions with the defined templates. </div>	
 Sort Headers	Changes the order in which the headers display. The available options are as follows: - By Importance – Displays first the important headers. - By name – Displays headers in alphabetical order.
 Show Headers	Chooses the headers to display. There are several types of headers—each displayed in a different color (see color codes). Choose Show Minimal or any combination of the following: <ul style="list-style-type: none"> • Mandatory • TCP Mandatory • Advisable • Required if SDP • Conditional • Optional • Invalid
 Preview	Shows the SIP message that is generated by using the current configuration.
Create	Saves the current message configuration in a file. If saved for the first time, a window prompts you to enter the file name. By default, the name includes the time and date at that moment.
Load	Loads an existing message configuration.
Save	Saves the current message configuration in a different file.
OK	Creates the ISP message and closes the Structured Message window.
Cancel	Discards the changes and closes the window.


***APPENDIX B* The Expression Evaluator Syntax**

The SetVar and TestVar script functions of the Flow test library enable you to define complex expressions – based on numerals, variables, operators, functions, and variables – that are evaluated at test execution time and activate different function outputs depending on the result of the evaluations.

This appendix described the entities supported by the Expression Evaluator that is part of these script functions.

Data Types

The supported data types are listed in the following table:

Type	Examples	Notes
Number	12 33.4523	Numbers can be integers or floating-point numbers.
String	"Welcome to the Voice Mail System"	Strings are always written between inverted commas.
Date	You can create data objects by using the <code>str2date("15-oct-01 14:32")</code> function or you get date objects by reading them from a database record: <code>\$Recordset1.DateTime</code>	To obtain the current date, use the <code>GetCurrentDate()</code> function. To get the components of a date (day, month, hours, and so on) you can use the <code>dtGet___</code> functions. For further information, see Functions .  Note: The date string format depends on the regional settings.

Operators

Supported operators—arithmetic, relational, logical, and format operators—are listed in the following tables:


Arithmetic Operators

Operator	Description	Usage	Examples	Notes
+	Addition	Number+ Number String + String String + Number Date + Number Date + String	20 + 5 =25 "Channel: " + \$MapPos is assessed as "Channel: 12" (\$MapPos is a system variable and its value is equal to the logical channel index).	For numbers, it performs arithmetic addition. For strings, it performs concatenation. Any operation involving strings gives a new string (concatenation). The number added to a date represents the number of days (or fraction of days —3.5 days is 3 days and 12 hours—). The result is a new date.
-	Subtraction	Number- Number Date - Date Date - Number	20 - 5 =15 str2date("2-nov- 01") - 1 is assessed as 1 nov 2001 (date) str2date("2-nov- 01") - str2date("1-nov-01") = 1	Arithmetic subtraction. The difference between two dates is a number of days. "Date - number" gives a new date.
*	Multiplication	Number * Number	20 * 5 = 100	Arithmetic multiplication
/	Division	Number / Number	20/5 = 4	Arithmetic division
%	Modulo-Division	Number%Number	20% 3 = 2	a% b = the remainder of the a/b division
^	Power	Number ^ Number	2 ^ 3 = 8	Raise to the given power

Relational Operators

Name	Description	Usage	Examples	Description
------	-------------	-------	----------	-------------

Appendix B The Expression Evaluator Syntax

<	Less Than	Number < Number String < String Date < Date	5 < 7 is assessed as 1 (true). "abc" < "axx" means 1 (true).	For strings, it performs the lexicographic comparison.  Note: To compare numbers and strings, use the str2num or num2str function.
>	Greater Than	Number > Number String > String Date > Date	5 > 7 is assessed as 0 (false). "John" > "Alex" means 1 (true).	For strings, it performs the lexicographic comparison.
<=	Less Or Equal	Number <= Number String <= String Date <= Date	5 <= 5.2 is evaluated as 1 (true). "Abc" <= "Abcd" is assessed as 1 (true).	For strings, it performs the lexicographic comparison.
>=	Greater Or Equal	Number >= Number String >= String Date >= Date	5 >= 5.2 is assessed as 0 (false). "XYZ" >= "ABC" is assessed as 1 (true).	For strings, it performs the lexicographic comparison.
==	Equal	Number == Number String == String Date == Date	5 == 5.2 is assessed as 0 (false). "XYZ" == "XY" + "Z" is assessed as 1 (true).	For strings, it performs the lexicographic comparison.
!=	Different	Number != Number String != String Date != Date	5 != 5.2 is assessed as 1 (false). "XYZ" != "xy" + "Z" is assessed as 1 (true).	For strings, it performs the lexicographic comparison.

Logical Operators

Operator	Description	Usage	Examples	Notes
----------	-------------	-------	----------	-------

&&	AND	Number && Number	(6 - 5 == 1) && (7 > 2) is assessed as 1 (true).	0 && 0 is assessed as 0. 1 && 0 is assessed as 0. 0 && 1 is assessed as 0. 1 && 1 is assessed as 1.
	OR	Number Number	("abc" == "a") (4 > 3) is assessed as 1 (true).	0 0 is assessed as 0. 1 0 is assessed as 1. 0 1 is assessed as 1. 1 1 is assessed as 1.
!	NOT	! Number (unary operator)	! ("abc" == "a") is assessed as 1 (true).	! 0 is assessed as 1. ! 1 is assessed as 0.
? :	Conditional Operator	Number ? (Number or String or Date) : (Number or String or Date)	(\$MapPos % 2 == 0) ? "Even" : "Odd" is equivalent with: If the channel index is an even number, (remainder upon division by 2 is zero) the expression evaluates as "Even"; otherwise, it evaluates as "Odd."	Condition? Expression1: Expression2. If the condition evaluates as true, then Expression1 is evaluated; otherwise, Expression2 is evaluated.

 **Note:** For all logical operations, 1 means true and 0 means false.

Format Operator

Format – Configures the output format of another variable or string of characters.

The behavior of the **Format** operator is similar with the output format from the Visual C printf function.

The syntax is: format("%TypeFieldCharacter", string).

 **Note:** The Format operator returns the character associated to the ASCII code.

The available TypeFieldCharacters are described in the following table:

Character	Type	Output Format
-----------	------	---------------

Appendix B The Expression Evaluator Syntax

C	Int	Character
C	Int or Wint_t	When used with the printf functions, it specifies a wide character; when used with the wprintf functions, it specifies a single-byte character.
D	Int	Signed decimal integer
I	Int	Signed decimal integer
O	Int	Unsigned octal integer
U	Int	Unsigned decimal integer
X	Int	Unsigned hexadecimal integer using "abcdef"
X	Int	Unsigned hexadecimal integer using "ABCDEF"
E	Double	Signed value having the [-]d.dddd e [sign]ddd form, where d is a single decimal digit, dddd is one or more decimal digits, ddd is exactly three decimal digits, and the sign is + or -.
E	Double	Identical to the e format, except that E rather than e introduces the exponent.
F	Double	Signed value having the [-]dddd.dddd form, where dddd is one or more decimal digits. The number of digits before the decimal point depends on the magnitude of the number, and the number of digits after the decimal point depends on the requested precision.
G	Double	Signed value printed in f or e format, whichever is more compact for the given value and precision. The e format is used only when the exponent of the value is less than -4 or greater than or equal to the precision argument. Trailing zeros are truncated, and the decimal point appears only if one or more digits follow.
G	Double	Identical to the g format, except that E, rather than e, introduces the exponent (where appropriate).
N	Pointer To Integer	Number of characters successfully written so far to the stream or buffer; this value is stored in the integer whose address is given as the argument.
P	Pointer To Void	Prints the address pointed to by the argument in the xxxx:yyyy format, where xxxx is the segment and yyyy is the offset, and the x and y digits are the uppercase hexadecimal digits.

S	String	When used with printf functions, it specifies a single-byte character string; when used with wprintf functions, it specifies a wide-character string. Characters are printed up to the first null character or until the precision value is reached.
S	String	When used with printf functions, it specifies a wide-character string; when used with wprintf functions, it specifies a single-byte-character string. Characters are printed up to the first null character or until the precision value is reached.

Functions

Supported functions are listed in the following table:

Function	Description	Usage	Examples
str2num	Converts a string to a number. The string should represent a real number (you cannot convert "abc" to a number, but you can convert "100" to the number 100).	str2num(string)	str2num("3"+"5") = 35
num2str	Converts a number to a string.	num2str (number)	num2str(3 + 5) = "8"
length	Calculates a string length.	length(string)	Set Variable \$string_ var == "abc" length (\$string_ var) = 3
extract	Extracts a text substring from a string.	extract (string,start_ position, end_ position)	Set Variable \$string_ var == "abcdefgh" extract(\$string_ var,2,5) = "bcde"
lookup	Searches for a a text in a string and returns 1 (true) if the text is found and 0 (false) otherwise.	lookup(string, text)	lookup(\$SIP_ Message,"TO:")
tick	Stores the time elapsed from the last restart of the machine in a user-defined variable. It is used for time difference measurements.	tick(void)	tick()
hrtick	Stores the time elapsed from the last restart of the machine in a user-defined variable. It is used for time difference measurements with high resolution. As compared to tick, it has a 12 millisecond (ms) resolution.	hrtick(void)	hrtick()

random	Returns a random number in the specified range.	random(min_number,max_number)	random(31,75) =54
generate guid	Returns a globally unique identifier (GUID) with a specified prefix and length. If no length is specified, the endpoint's MAC address string is appended to the prefix.	generateguid (prefix, guid_len)	generateguid ("qwe12",13)

Compound Variables

A compound variable contains a number of internal fields.

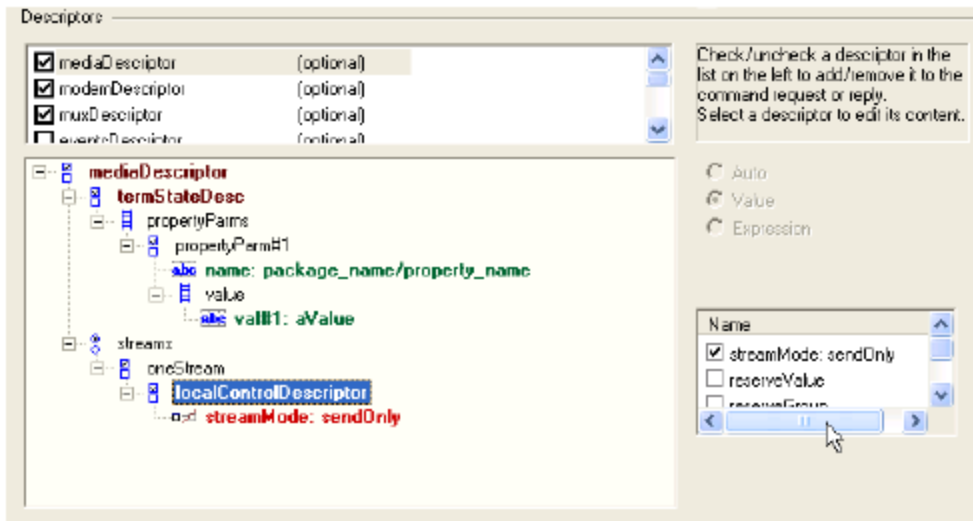
***APPENDIX C* Using the H248 Descriptive Editor**

The following sections help you configure descriptors for H.248/MEGACO commands by using the Descriptor Editor.

The Descriptive Editor GUI

An H.248 descriptor is an entity within a command carrying parameters related to a specific function of the protocol.

For each H.248 command, the Descriptors pane displays a list of supported descriptors. When a descriptor is selected in the list, it becomes the currently selected descriptor and its structure is displayed in the Descriptor Editor by using a tree representation.



The tree structure can contain both non-leaf and leaf (terminal) items.



Depending on the node selected at the tree level, different options are enabled or disabled. For non-leaf items, a list with subcomponents is displayed, while for leaf items an edit field becomes available enabling you to change the value.

Editing Transmitted Request Messages


For a transmitted request message, for each contained message descriptor its descriptor tree is edited for both non-leaf and leaf nodes.

Non-Leaf Items

A non-leaf item can be of the following types:

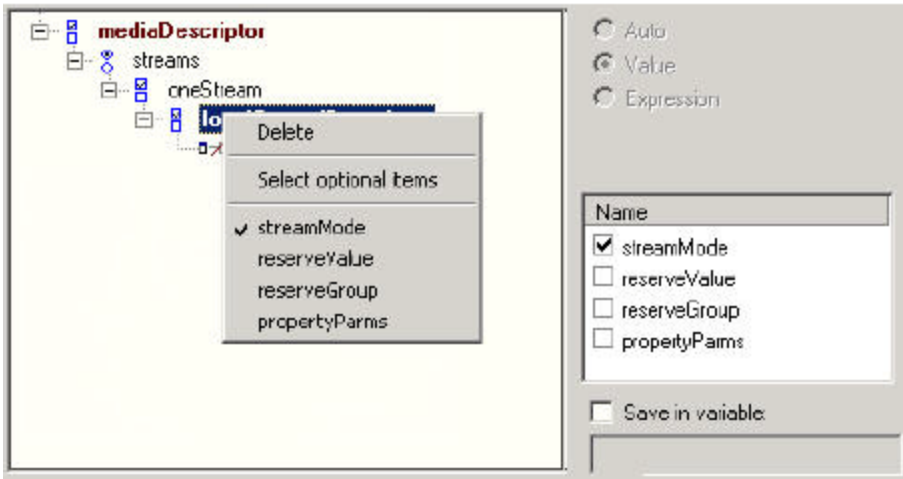
- Sequence  – This is a collection of sub-items, whereby each sub-item may be mandatory or optional. For an item of type sequence, if the item has optional sub-items, these will have a check box attached in the list with subcomponents to enable/disable them. An example is the localControlDescriptor in the mediaDescriptor.
- Sequenceof  – This is a set of subitems of the same type (for example, propertyParms in

localControlDescriptor).

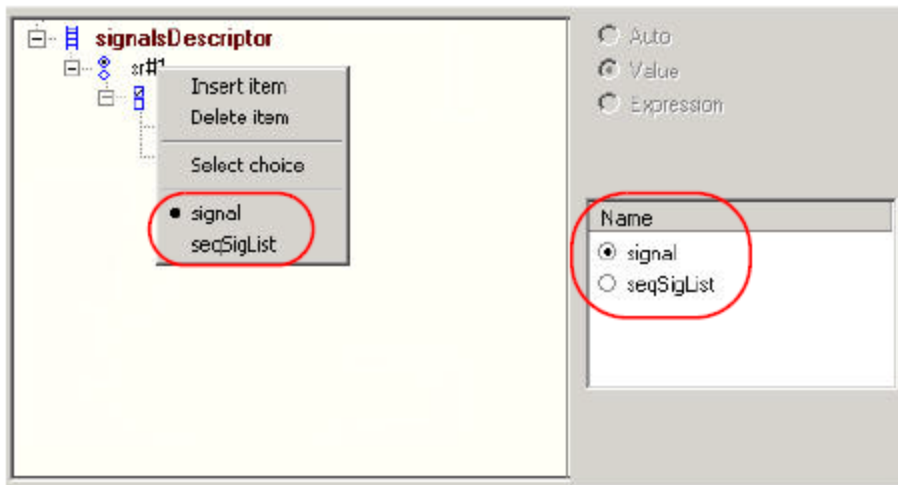
- Choice  – This a single sub-item that is selected from a subitems set.

To edit a non-leaf item, do the following::

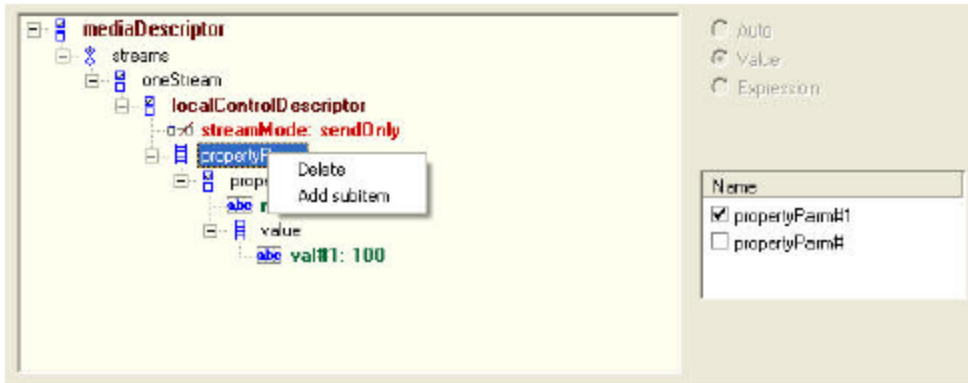
1. Select a node in the tree representation and activate optional subitems by right-clicking the node. A context menu shows the list of the node's subitems.
2. For an item of type sequence, click to check/uncheck the subitems in the Name pane at the right. The selected subitems are added to the tree list and the Name section updates to reflect the selection status.







For an item of type choice, the list with subitems has the possible choices under the Name section. Same choices are also available on the popup menu by right clicking the tree item.



For an item of type sequenceof, all existing subitems are available under the subcomponent list with a check option. The last entry is unchecked and used to add new subitems. Unchecking a sub-item deletes it.



Leaf Items

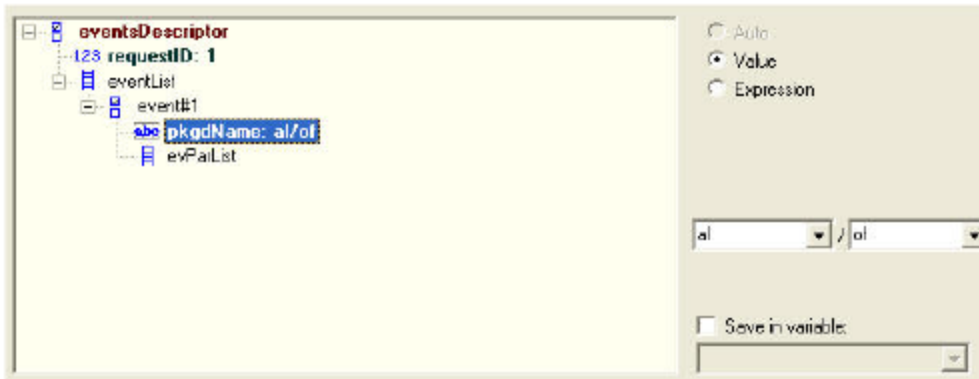
Leaf items may be of type string , number , boolean , or enum . The following options are available:

- Auto - If selected, this option sets an activity-level automatic value for the parameter at run-time. This option is available only for some elements (for example, RequestID in the Events or localDescriptor – SDP in the Media descriptor).
- Value - Turn on this option to specify a value for the message element. When choosing this option, an edit box becomes available near the option buttons.
- Expression - Turn on this option to specify a run-time evaluated expression for the message element. The expression creation rules are similar with those for expressions in other script functions, for example, a simple variable can be referred to as \$variable_name, a global variable can be used as \$array_var[4].

When choosing this option, an edit box becomes available near the option buttons.



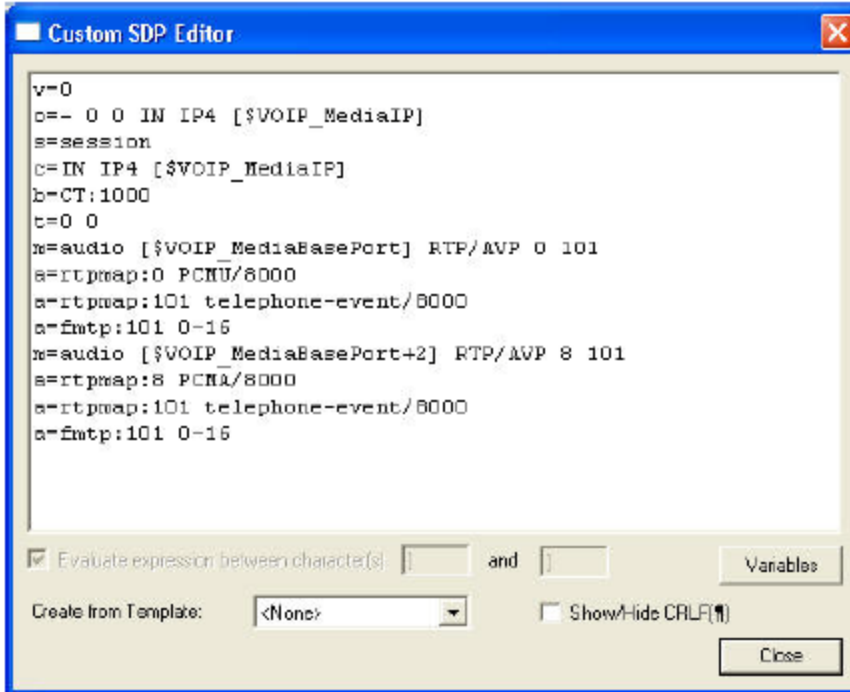
For package components, a value is edited as package name and event/signal/property name. Each name is edited in its own combo box, each combo box having a list of names from pre-loaded packages.



For SDP items, click **Edit SDF** to access the **Custom SDP** window.



In the **Custom SDP Editor** window, the SDP string can be edited. Several templates are also available. To select a template, click **Create from Template**.

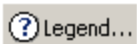


The values for leaf items may be saved into user-defined test scenario variables, for example, for referencing them in other script functions.

To save a sub-item value, click **Save in variable** and choose the name of the scenario variable in the list.



While in the Descriptor Editor, a legend of all node representations can be obtained by clicking the



button, which opens a window such as the following:



Editing Expected Response Messages

This operation is largely similar with the editing of transmitted H.248 request messages, in the sense that expected message elements are also specified by using the descriptors tree representation shown in [Descriptor Editor GUI](#).

An expected message item can be specified using one of the following options:

- Auto - If selected, this option sets an activity-level automatic value for the parameter at run-time. This option is available only for some elements (for example, terminationName or contextID).
- Match Value - If selected, the expected message element must have a specific value that is specified in the pane below.
- Match Expression - If selected, the expected message element is specified by using a run-time evaluated expression. The expression creation rules are similar with those for expressions in other script functions, for example, a simple variable can be referred to as \$variable_name, a global variable can be used as \$array_var[4].
- Match any value: If selected, the expected message element can have any value.
- Custom SDP string: If selected, the expected SDP definition needs specified in the **Custom SDP Editor** window that is accessed by clicking **Edit SDP**.
- Not allowed: If selected, the expected message element cannot be present in the expected reply.

This page intentionally left blank.

***APPENDIX D* Using the MGCP Parameter Editor**

The following sections provide information to configure the MGCP script function.

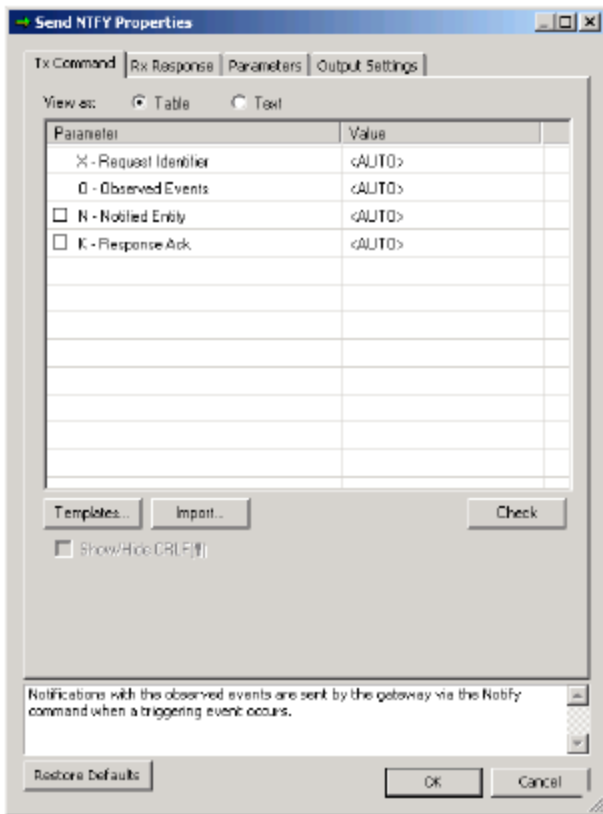
MGCP Script Functions Overview

IxLoad MGCP script functions fall into the following categories:

- Send type: These functions implement transactions that consist in sending an MGCP command followed by the receiving of an awaited response message. Such a function sends a command and then waits for any of the specified response messages.
- Wait type: These functions implement MGCP transactions that consist in the receiving of the specified command followed by the sending of a configured response message. Such a function waits for a matching MGCP command with matching parameters and then sends the configured response message.

Because both function types implement a command/response transaction model, for each script function, two different configuration pages are available, one for commands and another one for responses.

For example, in the case of the Send NOTIFY script function shown in the following image, the TX Command page is used to specify the Notify command's parameters, while in the RX Response page, you specify the response message an MGW entity waits for after having sent an NOTIFY command.



Script functions can be edited by manually editing their parameters, or by importing the commands from a text file.

Functions can also be created starting from templates.

Note: Commands that create or modify connections – CRCX, MDCX – also require you to specify an SDP definition that represents an endpoint’s media capabilities. Specifying an SDP is done by either choosing the use of the activity-level settings, or by manually editing the SDP in a separate SDP editor window.

Send Type Functions

Send-type script functions contain a page to specify the parameters of the sent MGCP command ([Tx Command Page](#)) and another one to specify the expected response message ([Rx Response Page](#)).

Tx Command Page

This page is used to edit the parameters of the sent MGCP command.

Any such page contains a command-specific list of mandatory and optional parameters. In the following image that shows the **Send NTFY** command in the Table viewing mode of the editor, the first two parameters (X, O) are mandatory, while the subsequent ones, N and K – prefixed by the selection control – are optional.

Parameter	Value
<input checked="" type="checkbox"/> X - Request Identifier	<AUTO>
<input checked="" type="checkbox"/> O - Observed Event	<AUTO>
<input type="checkbox"/> N - Notified Entity	<AUTO>
<input type="checkbox"/> K - Response Ack	<AUTO>

When choosing the Text viewing mode, the editor shows the command followed by the mandatory parameters only.

```
NTFY <TRAN_ID> <ENDPOINT> <MGCP_VER>
X: <AUTO>
O: <AUTO>
```

Note: While in the Text viewing modes, you can show/hide special characters by selecting/unselecting the Show/hide CRLF option.

Editing a command parameter is done by clicking the corresponding entry in the Value column and selecting the required value – a string or an IxLoad VoIP variable

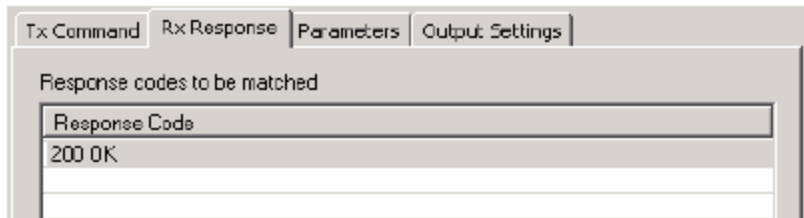
– instead of the AUTO value that is configured by default.

Note: The Auto value specifies an automatic activity-level value for the parameter that is assigned at run-time.

Note: Both editor views support the use of script variables and mathematic expressions comprising variables. Whenever variables are used, these need to be enclosed in "<" and ">", for example, in <\$CallAgentName>.

Rx Response Page

Editing a response is done in the Rx Response/Tx Response page, depending on whether the script function is of the Sent or the Wait type. By default, the response page is initially configured by using a 200 OK response.



You can add more awaited responses by clicking **Add** and choosing a response from the window that appears. When another response is added to the list of responses to be matched, an additional output is added to the script function.

Note: For example, assuming the Send NOTIFY function was configured in the **RX Response** tab by using the 200 OK and the 202 Accepted responses, after sending a NOTIFY command, the MGW would wait for either a 200 OK or a 202 Accepted response message.

To remove a response from the list, click **Delete**.

Wait Type Functions

Wait-type script functions contain a page to specify the expected MGCP command and parameters ([Tx Command Page](#)) and another one to specify the response message ([Rx Response Page](#)) that is sent when a matching command is received.

Rx Command Page

This page, used for specifying the parameters of the expected MGCP command, contains a command-specific list of mandatory and optional parameters, similar to that described in [Tx Command Page](#).

In addition to the command parameter to be matched, script functions that implement media-related functionality – Wait CRCX and Wait MDCX – contain the following additional options:

- Ignore SDP: If selected, the SDP definition contained in the incoming MGCP command is ignored.
- Extract SDP: If selected, the SDP definition contained in the incoming MGCP command is processed and media capability information is extracted.

Tx Response Page

When an MGCP command matching the definition from the Rx Command page is received, this page is used for specifying an MGCP response message.

By default, the sent response message is an automatically generated message based on the type of the received command and the current state of the MGCP entity receiving the message (Auto response option selected). Alternatively, when de-selecting the Auto response option, you can specify the sent response message by selecting it from a list and edit its parameters, as described in [Tx Command Page](#).

In addition to the command parameter to be matched, for script functions that implement media - related functionality – Wait CRCX and Wait MDCX – by default, the response also contains an SDP definition with activity-level settings (**Use activity settings** option is selected).

Alternatively, you can edit the sent SDP definition by clicking **Custom SDP**, clicking the

A rectangular button with a light gray background and a thin border, containing the text "Edit SDP body..." in a small, dark font.

button, and editing the definition manually in the editor window that appears.

This page intentionally left blank.

APPENDIX E Skinny Sample Configurations Overview

The following table provides an overview of configuration parameters for the predefined Skinny test samples:

Range of Device Name	Range of Phone Numbers	Min # of unique phones required	Phone Type	Group of Test Cases	Feature Settings
SK_001_7902_SO_US_100_Chs_IPv4_Static_Seq_Registration_5_retries					
SEP7902A00[00001-]	19[00001-]	100	7902	Bulk Registration	
SK_002_7960_SO_US_100_Chs_IPv4_Static_Seq_Registration_5_retries					
SEP7960A00[00001-]	16[00001-]	100	7960	Bulk Registration	
SK_005_7902_SO_US_100_Chs_IPv4_Static_Seq_Bulk_Registration_loop					
SEP7902A00[00001-]	19[00001-]	100	7902	Bulk Registration	
SK_006_7960_SO_US_100_Chs_IPv4_Static_Seq_Bulk_Registration_loop					
SEP7960A00[00001-]	16[00001-]	100	7960	Bulk Registration	
SK_003_7902_SO_US_100_Chs_IPv4_Static_Seq_Bulk_Registration_loop_5_retries					
SEP7902A00[00001-]	19[00001-]	100	7902	Bulk Registration	
SK_004_7960_SO_US_100_Chs_IPv4_Static_Seq_Bulk_Registration_loop_5_retries					
SEP7960A00[00001-]	16[00001-]	100	7960	Bulk Registration	
SK_007_7902_SO_US_100_Chs_IPv4_Static_Seq_Reg_5s_Sleep_Dereg_5_retries					
SEP7902A00[00001-]	19[00001-]	100	7902	Bulk Registration	
SK_008_7960_SO_US_100_Chs_IPv4_Static_Seq_Reg_5s_Sleep_Dereg_5_retries					
SEP7960A00[00001-]	16[00001-]	100	7960	Bulk Registration	

**SK_009_7902_SO_US_5000_ChS_IPv4_Static_Basic_Call_10s,
SK_010_7902_SO_US_5000_ChS_IPv4_Static_Basic_Call_3min,
SK_011_7902_SO_US_5000_ChS_IPv4_Static_Basic_Call_30min**

SEP7902A00[00001-] 19[00001-] 5000 7902 Bulk Call Testing

SEP7902B00[00001-] 31[00001-] 5000 7902 Bulk Call Testing

**SK_012_7902_SM_US_900_ChS_IPv4_Static_Basic_Call_Voice_10s,
SK_013_7902_SM_US_900_ChS_IPv4_Static_Basic_Call_Voice_3min,
SK_014_7902_SM_US_900_ChS_IPv4_Static_Basic_Call_Voice_30min**

SEP7902A00[00001-] 19[00001-] 900 7902 Bulk Call Testing

SEP7902B00[00001-] 31[00001-] 900 7902 Bulk Call Testing

**SK_016_7902_SM_US_300_ChS_IPv4_Static_Basic_Call_DTMFs_inband_3min
SK_017_7902_SM_US_300_ChS_IPv4_Static_Basic_Call_DTMFs_out-of-band_3min**

SEP7902A00[00001-] 19[00001-] 300 7902 Bulk Call Testing

SEP7902B00[00001-] 31[00001-] 300 7902 Bulk Call Testing

**SK_019_7902_SM_US_300_ChS_IPv4_Static_BasicCall_TONE_lowFreq_inband_3min
SK_020_7902_SM_US_300_ChS_IPv4_Static_BasicCall_TONE_medFreq_inband_3min
SK_021_7902_SM_US_300_ChS_IPv4_Static_BasicCall_TONE_highFreq_inband_3min**

SEP7902A00[00001-] 19[00001-] 5000 7902 Bulk Call Testing

SEP7902B00[00001-] 31[00001-] 5000 7902 Bulk Call Testing

SK_015_7902_SM_US_10K_BHCA_IPv4_Static_Basic_Call_Voice_1min

SEP7902A00[00001-] 19[00001-] 125 7902 Bulk Call Testing

SEP7902B00[00001-] 31[00001-] 125 7902 Bulk Call Testing

SK_018_7902_SM_US_25K_BHCA_IPv4_Static_Basic_Call_DTMFs_inband_3_min

SEP7902A00[00001-] 19[00001-] 900 7902 Bulk Call Testing

SEP7902B00[00001-] 31[00001-] 900 7902 Bulk Call Testing

SK_022_7960_SM_US_5_ChS_IPv4_Static_Hold_Resume

SEP7960AF0[00001-] 36[00001-] 5 7960 Call Features

SEP7960BF0[00001-] 37[00001-] 5 7960 Call Features

SK_031_7960_SM_US_5_Chs_IPv4_Static_List_Ad_Hoc_Conference

SEP7960AF0[00001-] 36[00001-] 5 7960 Call Features

SEP7960BF0[00001-] 37[00001-] 5 7960 Call Features

SEP7960CF0[00001-] 38[00001-] 5 7960 Call Features

SK_032_7960_SM_US_5_Chs_IPv4_Static_Forward_All_Calls

Appendix E Skinny Sample Configurations Overview

SEP7960AF0[00001-]	36[00001-]	5	7960	Call Features	
SEP7960BF0[00001-]	37[00001-]	5	7960	Call Features	
SEP7960CF0[00001-]	38[00001-]	5	7960	Call Features	
SK_033_7960_SM_US_5_Chs_IPv4_Static_Forward_Busy					
SEP7960AF4[00001-]	48[00001-]	5	7960	Call Features	
SEP7960BF4[00001-]	49[00001-]	5	7960	Call Features	Forward Busy Destination D (51[00001-]), Busy Trigger 1
SEP7960CF4[00001-]	50[00001-]	5	7960	Call Features	
SEP7960DF4[00001-]	51[00001-]	5	7960	Call Features	
SK_034_7960_SM_US_5_Chs_IPv4_Static_Forward_No_Answer					
SEP7960AF5[00001-]	52[00001-]	5	7960	Call Features	
SEP7960BF5[00001-]	53[00001-]	5	7960	Call Features	Forward No Answer Destination C (54[00001-])
SEP7960CF5[00001-]	54[00001-]	5	7960	Call Features	
SK_026_7960_SM_US_5_Chs_IPv4_Static_Ad_hoc_Conference					
SEP7960AF0[00001-]	36[00001-]	5	7960	Call Features	
SEP7960BF0[00001-]	37[00001-]	5	7960	Call Features	
SEP7960CF0[00001-]	38[00001-]	5	7960	Call Features	
SK_027_7960_SM_US_5_Chs_IPv4_Static_MeetMe_Conference					
SEP7960AF0[00001-]	36[00001-]	5	7960	Call Features	
SEP7960BF0[00001-]	37[00001-]	5	7960	Call Features	
SEP7960CF0[00001-]	38[00001-]	5	7960	Call Features	
SK_028_7960_SM_US_5_Chs_IPv4_Static_Join_2_Calls					
SEP7960AF0[00001-]	36[00001-]	5	7960	Call Features	
SEP7960BF0[00001-]	37[00001-]	5	7960	Call Features	
SEP7960CF0[00001-]	38[00001-]	5	7960	Call Features	
SK_024_7960_SM_US_5_Chs_IPv4_Static_Blind_Transfer					
SEP7960AF0[00001-]	36[00001-]	5	7960	Call Features	

SEP7960BF0[00001-]	37[00001-]	5	7960	Call Features	
SEP7960CF0[00001-]	38[00001-]	5	7960	Call Features	
SK_023_7960_SM_US_5_ChS_IPv4_Static_Transfer_with_Consultation					
SEP7960AF0[00001-]	36[00001-]	5	7960	Call Features	
SEP7960BF0[00001-]	37[00001-]	5	7960	Call Features	
SEP7960CF0[00001-]	38[00001-]	5	7960	Call Features	
SK_025_7960_SM_US_5_ChS_IPv4_Static_Direct_Transfer_of_2_parties_on_a_line					
SEP7960AF0[00001-]	36[00001-]	5	7960	Call Features	
SEP7960BF0[00001-]	37[00001-]	5	7960	Call Features	
SEP7960CF0[00001-]	38[00001-]	5	7960	Call Features	
SK_036_7960_SM_US_5_ChS_IPv4_Static_Call_Group_Pickup					
SEP7960AF1[00001-]	39[00001-]	5	7960	Call Features	Call Group A (1330)
SEP7960BF1[00001-]	40[00001-]	5	7960	Call Features	Call Group B (1330)
SEP7960CF1[00001-]	41[00001-]	5	7960	Call Features	Call Group C (1331)
SK_037_7960_SM_US_5_ChS_IPv4_Static_Call_Pickup					
SEP7960AF2[00001-]	42[00001-]	5	7960	Call Features	Call Group A (1330)
SEP7960BF2[00001-]	43[00001-]	5	7960	Call Features	Call Group B (1330)
SEP7960CF2[00001-]	44[00001-]	5	7960	Call Features	Call Group C (1330)
MIX_023_7960_SO_US_3000_ChS_IPv4_Static_SK_to_SIP_Call_10s					
MIX_024_7960_SO_US_3000_ChS_IPv4_Static_SK_to_SIP_Call_3min					
MIX_025_7960_SO_US_3000_ChS_IPv4_Static_SK_to_SIP_Call_30_min					
SEP7960A00[00001-]	16[00001-]	3000	7960	Mixed SIP -Skinny, SIP UAs	
Note: SIP endpoints emulated by the VoIPSIPPeer test activity use registration names and phone numbers defined by the 7960BBBB[0000-] and 30[00001-] sequence generating expressions.					
MIX_026_7960_SO_US_3000_ChS_IPv4_Static_SIP_to_Sk_Call_10s					
MIX_027_7960_SO_US_3000_ChS_IPv4_Static_SIP_to_Sk_Call_3min					
MIX_028_7960_SO_US_3000_ChS_IPv4_Static_SIP_to_Sk_Call_30_min					
SEP7960A00[00001-]	16[00001-]	3000	7960	Mixed SIP -Skinny, SIP UAs	
Note: SIP endpoints emulated by the VoIPSIPPeer test activity use registration names and phone numbers defined by the 7960BBBB[0000-] and 30[00001-] sequence generating expressions.					

MIX_020_7960_SM_US_900_Chs_IPv4_Static_SIP_to_Sk_Call_Voice_10s
MIX_021_7960_SM_US_900_Chs_IPv4_Static_SIP_to_Sk_Call_Voice_3min
MIX_022_7960_SM_US_900_Chs_IPv4_Static_SIP_to_Sk_Call_Voice_30min

SEP7960A00[00001-] 16[00001-] 900 7960 Mixed SIP -Skinny,
 SIP UAs

Note: SIP endpoints emulated by the VoIPSIPPeer test activity use registration names and phone numbers defined by the *7960BBBB[0000-]* and *30[00001-]* sequence generating expressions.

MIX_017_7960_SM_US_900_Chs_IPv4_Static_Sk_to_SIP_Call_Voice_10s
MIX_018_7960_SM_US_900_Chs_IPv4_Static_Sk_to_SIP_Call_Voice_3min
MIX_019_7960_SM_US_900_Chs_IPv4_Static_Sk_to_SIP_Call_Voice_30min

SEP7960A00[00001-] 16[00001-] 900 7960 Mixed SIP -Skinny,
 SIP UAs

Note: SIP endpoints emulated by the VoIPSIPPeer test activity use registration names and phone numbers defined by the *7960BBBB[0000-]* and *30[00001-]* sequence generating expressions.

MIX_031_7960_SO_US_75k_BHCA_IPv4_Static_Sk_to_SIP_Call
MIX_029_7960_SM_US_75k_BHCA_IPv4_Static_Sk_to_SIP_Call_Voice

SEP7960A00[00001-] 16[00001-] 3000/900 7960 Mixed SIP -Skinny,
 SIP UAs

Note: SIP endpoints emulated by the VoIPSIPPeer test activity use registration names and phone numbers defined by the *7960BBBB[0000-]* and *30[00001-]* sequence generating expressions.

MIX_032_7960_SO_US_75k_BHCA_IPv4_Static_SIP_to_Sk_Call
MIX_030_7960_SM_US_75k_BHCA_IPv4_Static_SIP_to_Sk_Call_Voice

SEP7960A00[00001-] 16[00001-] 3000/900 7960 Mixed SIP -Skinny,
 SIP UAs

Note: SIP endpoints emulated by the VoIPSIPPeer test activity use registration names and phone numbers defined by the *7960BBBB[0000-]* and *30[00001-]* sequence generating expressions.

MIX_001_7960_SO_US_5k_Chs_IPv4_Static_SK_to_SIP_trunk_Bulk_Call_10s
MIX_002_7960_SO_US_5k_Chs_IPv4_Static_SK_to_SIP_trunk_Bulk_Call_3min
MIX_003_7960_SO_US_5k_Chs_IPv4_Static_SK_to_SIP_trunk_Bulk_Call_30min

SEP7960A00[00001-] 16[00001-] 5000 7960 Mixed SIP -Skinny,
 SIP Trunk

Note: The SIP endpoints emulated by the VoIPSIPPeer test activity use phone numbers defined by the *20000[00001-]* sequence generating expression.

MIX_004_7960_SO_US_5k_Chs_IPv4_Static_SIP_to_SK_trunk_Bulk_Call_10s
MIX_005_7960_SO_US_5k_Chs_IPv4_Static_SIP_to_SK_trunk_Bulk_Call_3min
MIX_006_7960_SO_US_5k_Chs_IPv4_Static_SIP_to_SK_trunk_Bulk_Call_30min

SEP7960A00[00001-] 16[00001-] 5000 7960 Mixed SIP -Skinny,
 SIP Trunk

Note: The SIP endpoints emulated by the VoIPSIPPeer test activity use phone numbers defined by the 20000[00001-] sequence generating expression.

MIX_012_7960_SM_US_900_ChS_IPv4_Static_SIP_SK_trunk_Call_Voice_10s
MIX_013_7960_SM_US_900_ChS_IPv4_Static_SIP_SK_trunk_Call_Voice_3min
MIX_014_7960_SM_US_900_ChS_IPv4_Static_SIP_SK_trunk_Call_Voice_30min

SEP7960A00[00001-] 16[00001-] 900 7960 Mixed SIP -Skinny,
SIP Trunk

Note: The SIP endpoints emulated by the VoIPSIPPeer test activity use phone numbers defined by the 20000[00001-] sequence generating expression.

MIX_009_7960_SM_US_900_ChS_IPv4_Static_SK_to_SIP_trunk_Call_Voice_10s
MIX_010_7960_SM_US_900_ChS_IPv4_Static_SK_to_SIP_trunk_Call_Voice_3min
MIX_011_7960_SM_US_900_ChS_IPv4_Static_SK_to_SIP_trunk_Call_Voice_30min

SEP7960A00[00001-] 16[00001-] 900 7960 Mixed SIP -Skinny,
SIP Trunk

Note: The SIP endpoints emulated by the VoIPSIPPeer test activity use phone numbers defined by the 20000[00001-] sequence generating expression.

MIX_008_7960_SO_US_80k_BHCA_IPv4_Static_SIP_to_SK_trunk_Bulk_Call

SEP7960A00[00001-] 16[00001-] 5000 /900 7960 Mixed SIP -Skinny,
SIP Trunk

Note: The SIP endpoints emulated by the VoIPSIPPeer test activity use phone numbers defined by the 20000[00001-] sequence generating expression.

MIX_016_7960_SM_US_75k_BHCA_IPv4_Static_SIP_to_SK_trunk_Call_Voice

SEP7960A00[00001-] 16[00001-] 5000 /900 7960 Mixed SIP -Skinny,
SIP Trunk

Note: The SIP trunk endpoints by the VoIPSIPPeer test activity use phone numbers defined by the 20000[00001-] sequence generating expression.

MIX_007_7960_SO_US_80k_BHCA_IPv4_Static_SK_to_SIP_trunk_Bulk_Call

SEP7960A00[00001-] 16[00001-] 5000 /900 7960 Mixed SIP -Skinny,
SIP Trunk

Note: The SIP endpoints emulated by the VoIPSIPPeer test activity use phone numbers defined by the 20000[00001-] sequence generating expression.

MIX_015_7960_SM_US_75k_BHCA_IPv4_Static_SK_to_SIP_trunk_Call_Voice

SEP7960A00[00001-] 16[00001-] 5000 /900 7960 Mixed SIP -Skinny,
SIP Trunk

Note: The SIP endpoints emulated by the VoIPSIPPeer test activity use phone numbers defined by the 20000[00001-] sequence generating expression.

This page intentionally left blank.

APPENDIX F Support for Multipart SIP Messages

The IxLoad Voice Plug-in supports SIP messages having multipart bodies, whereby each part is encoded using the MIME format. In addition to the most common application/sdp type, the Content-Type parameter supports the following multipart extensions:

- multipart/mixed: This extension is used for sending additional information to that contained in the SDP. The additional information can be an UE location, an XML text containing resources, and images.
- multipart/alternative: This extension is used for sending different versions (formats) of the same information.
- multipart/related: This extension is used for sending an object comprised of multiple related elements, for example a web page containing multiple images.

Whenever a multipart message body is created, the different parts have to be separated by the boundary parameter, and the body must also be terminated by using this separator string. For example, assuming we had a SIP message that contains both an sdp and an xml part, the Content-Type and the separator definition would be as shown in the following example:

```
INVITE sip:conf-fact@example.com SIP/2.0
Content-Type: multipart/mixed; boundary="boundary1"
Content-Length: 619
```

```
--boundary1
```

```
Content-Type: application/sdp
```

```
v=0
```

```
o=Alice 2890844526 2890842807 IN IP4 atlanta.example.com
```

```
s=-
```

```
c=IN IP4 192.0.2.1
```

```
t=0 0
```

```
m=audio 20000 RTP/AVP 0
```

```
a=rtpmap:0 PCMU/8000
```

```
m=video 20002 RTP/AVP 31
```

```
a=rtpmap:31 H261/90000
```

First part (SDP content)

```
--boundary1
```

```
Content-Type: application/resource-lists+xml
```

```
Content-Disposition: recipient-list
```

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<resource-lists xmlns="urn:ietf:params:xml:ns:resource-lists">
```

```
  <list>
```

```
    <entry uri="sip:bill@example.com"/>
```

```
    <entry uri="sip:randy@example.net"/>
```

```
    <entry uri="sip:joe@example.org"/>
```

```
  </list>
```

```
</resource-lists>
```

```
--boundary1--
```

Second part (XML content)

