

ONVIF[®]

Profile S Client Test Specification

Version 22.06

June 2022

© 2022 ONVIF, Inc. All rights reserved.

Recipients of this document may copy, distribute, publish, or display this document so long as this copyright notice, license and disclaimer are retained with all copies of the document. No license is granted to modify this document.

THIS DOCUMENT IS PROVIDED "AS IS," AND THE CORPORATION AND ITS MEMBERS AND THEIR AFFILIATES, MAKE NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THIS DOCUMENT ARE SUITABLE FOR ANY PURPOSE; OR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

IN NO EVENT WILL THE CORPORATION OR ITS MEMBERS OR THEIR AFFILIATES BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL, PUNITIVE OR CONSEQUENTIAL DAMAGES, ARISING OUT OF OR RELATING TO ANY USE OR DISTRIBUTION OF THIS DOCUMENT, WHETHER OR NOT (1) THE CORPORATION, MEMBERS OR THEIR AFFILIATES HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES, OR (2) SUCH DAMAGES WERE REASONABLY FORESEEABLE, AND ARISING OUT OF OR RELATING TO ANY USE OR DISTRIBUTION OF THIS DOCUMENT. THE FOREGOING DISCLAIMER AND LIMITATION ON LIABILITY DO NOT APPLY TO, INVALIDATE, OR LIMIT REPRESENTATIONS AND WARRANTIES MADE BY THE MEMBERS AND THEIR RESPECTIVE AFFILIATES TO THE CORPORATION AND OTHER MEMBERS IN CERTAIN WRITTEN POLICIES OF THE CORPORATION.

REVISION HISTORY

Vers.	Date	Description
22.06	Dec 27, 2022	Profile Normative Reference were removed from test cases according to #364
21.12	Oct 12, 2021	The following was done according to #425: Check Condition based on Device Features of Discovery feature was changed from 'All' to 'Discovery'
21.06	Jun 03, 2021	The following was updated according to #325: SETSYNCHRONIZATIONPOINT-1 test name was changed from SET SYNCHRONIZATION POINT to SET SYNCHRONIZATION POINT (EVENT SERVICE). Set Synchronization Point feature was renamed to Set Synchronization Point (Event Service)
20.12	Dec 8, 2020	NVTDISCOVERYTYPEFILTER-1 NVT DISCOVERY TYPE FILTER was updated according to #406: Types value check was updated to accept QName list instead of one QName value.
20.12	Nov 12, 2020	The following was done according to #399: System Date and Time Configuration: Check Condition based on Device Features was updated
20.12	Oct 27, 2020	The following was done according to #394: Check Condition based on Device Features of Network Configuration feature was changed from 'All' to 'Network Configuration'
20.12	Oct 27, 2020	The following was done according to #393: Check Condition based on Device Features of User Handling feature was changed from 'All' to 'User Configuration'
20.12	Aug 31, 2020	Set Synchronization Point Feature: Check Condition based on Device Features was changed according to #325.
20.12	Aug 31, 2020	Unsubscribe Feature: Check Condition based on Device Features was changed according to #325.
20.12	Aug 31, 2020	Keep Alive for Pull Point Event Handling Feature: Check Condition based on Device Features was changed according to #325.
20.12	Aug 31, 2020	Event Handling Feature: Check Condition based on Device Features was changed according to #325.
19.12	Oct 04, 2019	Note about not found GetStreamUri was added in the following test cases according to #339: MEDIASTREAMING-3 STREAMING OVER RTSP MEDIASTREAMING-4 STREAMING OVER UDP MEDIASTREAMING-5 STREAMING OVER HTTP VIDEOSTREAMING-1 MJPEG VIDEO STREAMING

		<p>VIDEOSTREAMING-2 MPEG4 VIDEO STREAMING</p> <p>VIDEOSTREAMING-3 H264 VIDEO STREAMING</p> <p>AUDIOSTREAMING-2 G.711</p> <p>AUDIOSTREAMING-3 G.726</p> <p>AUDIOSTREAMING-4 AAC</p>
19.12	Sep 18, 2019	<p>The following was done according to #325:</p> <p>Scope\Supplementary Features and Test Cases sections was added.</p> <p>Supplementary Features and Test Cases sections was added.</p>
19.12	Aug 13, 2019	<p>The following was done according to #325:</p> <p>EVENTHANDLING-3 METADATA STREAMING test was removed from Event Handling Feature and moved to Metadata Streaming Using Media2. Test case ID was changed to MEDIA2_METADATASTREAMING-1. Event Handling will use link to this test.</p> <p>EVENTHANDLING-4 METADATA STREAMING USING MEDIA was added for Profile S Devices.</p>
19.12	Sep 6, 2019	<p>NVTDISCOVERYTYPEFILTER-1 NVT DISCOVERY TYPE FILTER was updated according to #323:</p> <p>Unnecessary step with check that ProbeMatch is sent to Client IP address was removed.</p>
19.12	Aug 12, 2019	<p>The following was done according to #341:</p> <p>PTZ - Auxiliary Command section and PTZ - Auxiliary Command Test Cases section was moved from ONVIF PTZ Client Test Specification to ONVIF Profile S Client Test Specifications.</p>
19.12	Aug 12, 2019	<p>The following was done according to #341:</p> <p>PTZ Home Position section and PTZ Home Position Test Cases section was moved from ONVIF PTZ Client Test Specification to ONVIF Profile S Client Test Specifications.</p>
19.12	Aug 12, 2019	<p>The following was done according to #341:</p> <p>PTZ Presets section and PTZ Presets Test Cases section was moved from ONVIF PTZ Client Test Specification to ONVIF Profile S Client Test Specifications.</p>
19.12	Aug 12, 2019	<p>The following was done according to #341:</p> <p>PTZ Zoom Relative Positioning section and PTZ Zoom Relative Positioning Test Cases section was moved from ONVIF PTZ Client Test Specification to ONVIF Profile S Client Test Specifications.</p>
19.12	Aug 12, 2019	<p>The following was done according to #341:</p> <p>PTZ Pan Tilt Relative Positioning section and PTZ Pan Tilt Relative Positioning Test Cases section was moved from ONVIF PTZ Client Test Specification to ONVIF Profile S Client Test Specifications.</p>
19.12	Aug 12, 2019	<p>The following was done according to #341:</p>

		PTZ Zoom Absolute Positioning section and PTZ Zoom Absolute Positioning Test Cases section was moved from ONVIF PTZ Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: PTZ Pan Tilt Absolute Positioning section and PTZ Pan Tilt Absolute Positioning Test Cases section was moved from ONVIF PTZ Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: PTZ Zoom Continuous Positioning section and PTZ Zoom Continuous Positioning Test Cases section was moved from ONVIF PTZ Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: PTZ Pan Tilt Continuous Positioning section and PTZ Pan Tilt Continuous Positioning Test Cases section was moved from ONVIF PTZ Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: PTZ - Configuration section and PTZ - Configuration Test Cases section was moved from ONVIF PTZ Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: PTZ - Listing section and PTZ - Listing Test Cases section was moved from ONVIF PTZ Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: Network Protocols Configuration section and Network Protocols Configuration Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: DNS Configuration section and DNS Configuration Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: Hostname Configuration section and Hostname Configuration Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: System Date and Time Configuration section and System Date and Time Configuration Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341:

		IP Address Filtering section and IP Address Filtering Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: Zero Configuration section and Zero Configuration Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: Dynamic DNS section and Dynamic DNS Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: NTP section and NTP Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: Relay Outputs section and Relay Outputs Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: User Handling section and User Handling Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: System section and System Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: Network Configuration section and Network Configuration Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: Network Video Transmitter Discovery Type Filter section and Network Video Transmitter Discovery Type Filter Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: Discovery section and Discovery Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: Keep Alive for Pull Point Event Handling section and Keep Alive for Pull Point Event Handling Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341:

		Unsubscribe section and Unsubscribe Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: Set Synchronization Point section and Set Synchronization Point Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: Event Handling section and Event Handling Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: Capabilities section and Capabilities Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: HTTP Digest section and HTTP Digest Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.12	Aug 12, 2019	The following was done according to #341: Username Token section and Username Token Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile S Client Test Specifications.
19.06	Jun 14, 2019	The following was done according to #309: 'Validated Feature' section for each feature updated to be synchronized with feature ID used in feature list. 'Feature Under Test' section for each test case updated to be synchronized with sub-feature ID used in feature list. 'Validated Feature List' test case section removed.
19.06	Mar 28, 2019	The following was updated in the scope of #319: AUDIOSTREAMING-4 AAC AUDIO STREAMING (MP4A-LATM encoding name added)
18.06	Jun 21, 2018	Reformatting document using new template
18.06	Apr 05, 2018	'Required Number of Devices Summary' Annex added according to #241
18.06	Feb 16, 2018	The following were updated in the scope of #241: Feature Level Requirement (updated with new rules) Each Feature Level Requirement (updated with Check Condition based on Device Features and Required Number of Devices)
17.06	Jun 15, 2017	Links in Normative references section were updated.
17.06	Jun 06, 2017	The following PTZ test cases were moved into PTZ Client Test Specification according to #194: PTZ - Listing

		<p>PTZ - Configuration</p> <p>PTZ - Continuous Positioning</p> <p>PTZ - Absolute Positioning</p> <p>PTZ - Relative Positioning</p> <p>PTZ - Presets</p> <p>PTZ - Home Position</p> <p>PTZ - Auxiliary Command</p> <p>PTZ - Auxiliary Command</p>
17.06	May 05, 2017	<p>VIDEOENCODERCONFIGURATIONS-1 LIST VIDEO ENCODER CONFIGURATIONS was updated according to #197.</p> <p>MEDIAPROFILECONFIGURATIONS-1 LIST AVAILABLE MEDIA PROFILES was updated according to #198.</p> <p>MEDIASTREAMING-1 GET PROFILES was updated according to #198.</p> <p>MEDIAPROFILECONFIGURATIONS-3 CREATE A MEDIA PROFILE was updated according to #199.</p>
17.06	Apr 04, 2017	<p>Profile T Normative Reference added for the following features:</p> <p>PTZ - Presets Test Cases</p> <p>PTZ - Home Position Test Cases</p>
17.06	Mar 31, 2017	<p>The following test cases were updated according to #179:</p> <p>VIDEOSOURCECONFIGURATIONS-1 LIST VIDEO SOURCE CONFIGURATIONS</p> <p>MEDIASTREAMING-2 GET STREAM URI</p> <p>The following test cases were updated according to #168:</p> <p>MEDIASTREAMING-3 STREAMING OVER RTSP</p> <p>MEDIASTREAMING-4 STREAMING OVER UDP</p> <p>MEDIASTREAMING-5 STREAMING OVER HTTP</p> <p>VIDEOSTREAMING-1 MJPEG VIDEO STREAMING</p> <p>VIDEOSTREAMING-2 MPEG4 VIDEO STREAMING</p> <p>VIDEOSTREAMING-3 H264 VIDEO STREAMING</p> <p>MULTICASTSTREAMING-1 MULTICAST STREAMING USING RTSP</p> <p>AUDIOSTREAMING-2 G.711 AUDIO STREAMING</p> <p>AUDIOSTREAMING-3 G.726 AUDIO STREAMING</p> <p>AUDIOSTREAMING-4 AAC AUDIO STREAMING</p> <p>MULTIPLEVIDEOSOURCES-1 STREAMING WITH ALL VIDEO SOURCES DETECTED IN GET PROFILES</p>

		MULTIPLEAUDIOSOURCES-1 STREAMING WITH ALL AUDIO SOURCES DETECTED IN GET PROFILES
17.06	Mar 02, 2017	Media Profile Configurations Feature was updated according to #124 Profile S Normative Reference of GET SPECIFIC MEDIA PROFILE was changed to Optional according to #124
16.12	Dec 06, 2016	Test steps with check that RTSP SETUP, RTSP PLAY and RTSP TEARDOWN are not tunneled in HTTP were added in the following test case: VIDEOSTREAMING-3.
16.07	Jun 14, 2016	Test steps sequence was changed in the following test cases: VIDEOSTREAMING-1, VIDEOSTREAMING-2, VIDEOSTREAMING-3, AUDIOSTREAMING-2, AUDIOSTREAMING-3, AUDIOSTREAMING-4.
16.07	May 27, 2016	The following test case was updated: MULTICASTSTREAMING-2 MULTICAST STREAMING USING SOAP
16.07	Apr 18, 2016	Step description in Test Procedure was updated for the test cases: MEDIASTREAMING-3, MEDIASTREAMING-4, MEDIASTREAMING-5, VIDEOSTREAMING-1, VIDEOSTREAMING-2, VIDEOSTREAMING-3, MULTICASTSTREAMING-1, AUDIOSTREAMING-2, AUDIOSTREAMING-3, AUDIOSTREAMING-4, MULTIPLEVIDEOSOURCES-1, MULTIPLEAUDIOSOURCES-1 Old description: Device response has code RTSP 200 OK if it is detected New description: If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK
16.07	Apr 15, 2016	PTZ - Continuous Positioning scenario was updated PTZCONTINUOUSPOSITIONING-3 PTZ STOP test case was replaced by two test cases: PTZCONTINUOUSPOSITIONING-3 PTZ STOP and PTZCONTINUOUSPOSITIONING-4 STOP MOVEMENT USING PTZ CONTINUOUS MOVE New Pre-Requisite added for PTZCONTINUOUSPOSITIONING-1 PTZ CONTINUOUS MOVE PAN/TILT: Device supports PTZContinuousPanTilt New Pre-Requisite added for PTZCONTINUOUSPOSITIONING-2 PTZ CONTINUOUS MOVE ZOOM: Device supports PTZContinuousZoom NOTE was removed from PTZCONTINUOUSPOSITIONING-1 PTZ CONTINUOUS MOVE PAN/TILT NOTE was removed from PTZCONTINUOUSPOSITIONING-2 PTZ CONTINUOUS MOVE ZOOM
16.07	Mar 18, 2016	Checking of TEARDOWN response was changed in Test Procedure and PASS criteria for the test cases and annexes: MEDIASTREAMING-3, MEDIASTREAMING-4, MEDIASTREAMING-5, VIDEOSTREAMING-1, VIDEOSTREAMING-2, VIDEOSTREAMING-3, MULTICASTSTREAMING-1, AUDIOSTREAMING-2, AUDIOSTREAMING-3, AUDIOSTREAMING-4,

		<p>MULTIPLEVIDEOSOURCES-1, MULTIPLEAUDIOSOURCES-1, Annex A.3, Annex A.6</p> <p>Old description of checking of TEARDOWN response in Test Procedure:</p> <p>Device responds with code RTSP 200 OK.</p> <p>New description of checking of TEARDOWN response in Test Procedure:</p> <p>Device response has code RTSP 200 OK if it is detected.</p> <p>Old description of checking of TEARDOWN response in PASS criteria:</p> <p>Device response on the RTSP TEARDOWN request fulfills the following requirements:</p> <p>New description of checking of TEARDOWN response in PASS criteria:</p> <p>If there is Device response on the RTSP TEARDOWN request then it fulfills the following requirements:</p>
16.07	Mar 16, 2016	Docbook stylesheets were updated.
16.07	Mar 14, 2016	www.onvif.org was removed from Copyright section.
16.07	Mar 09, 2016	Minor changes: typos were fixed.
16.07	Feb 24, 2016	<p>Multiple Audio Sources Test Cases were added</p> <p>Annex A.4 Get Audio Sources List from GetProfiles responses was added</p> <p>Annex A.5 Get Audio Source Token That was Used for Streaming</p> <p>Annex A.6 Find Audio Streaming corresponding to GetStreamUri was added</p>
16.07	Feb 16, 2016	<p>Multiple Video Sources Test Cases were added</p> <p>Annex A.1 Get Video Sources List from GetProfiles was added</p> <p>Annex A.2 Get Video Source Token That was Used for Streaming was added</p> <p>Annex A.3 Find Video Streaming corresponding to GetStreamUri was added</p>
16.07	Feb 08, 2016	<p>Video Source Configurations Test Cases were updated: Profile S Requirement of LIST VIDEO SOURCE CONFIGURATIONS test was changed to Optional Profile S Requirement of GET SPECIFIC VIDEO SOURCE CONFIGURATION test was changed to Optional MODIFY VIDEO SOURCE CONFIGURATION test was split to tree tests: GET VIDEO SOURCE CONFIGURATION OPTIONS, SET VIDEO SOURCE CONFIGURATION and GET COMPATIBLE VIDEO SOURCE CONFIGURATIONS.</p> <p>Video Encoder Configurations Test Cases were updated: Profile S Requirement of LIST VIDEO ENCODER CONFIGURATIONS test was changed to Optional Profile S Requirement of GET SPECIFIC VIDEO ENCODER CONFIGURATION test was changed to Optional MODIFY VIDEO ENCODER CONFIGURATION test was split to two</p>

		tests: GET VIDEO ENCODER CONFIGURATION OPTIONS and SET VIDEO ENCODER CONFIGURATION.
16.07	Jan 26, 2016	<p>The description about structure and hierarchy was replaced for the test cases: MEDIASTREAMING-1, MEDIASTREAMING-2, MULTICASTSTREAMING-2, VIDEOENCODERCONFIGURATIONS-1, VIDEOENCODERCONFIGURATION-2, VIDEOENCODERCONFIGURATION-3, MEDIAPROFILECONFIGURATIONS-1, MEDIAPROFILECONFIGURATIONS-2, MEDIAPROFILECONFIGURATIONS-3, VIDEOSOURCECONFIGURATIONS-1, VIDEOSOURCECONFIGURATIONS-2, VIDEOSOURCECONFIGURATION-3, VIDEOSOURCECONFIGURATION-4, PTZLISTING-1, PTZLISTING-2, PTZCONFIGURATION-1, PTZCONFIGURATION-1, PTZCONTINUOUSPOSITIONING-1, PTZCONTINUOUSPOSITIONING-2, PTZABSOLUTEPOSITIONING-1, PTZABSOLUTEPOSITIONING-2, PTZRELATIVEPOSITIONING-1, PTZRELATIVEPOSITIONING-2, PTZPRESETS-1, PTZPRESETS-2, PTZHOMEPOSITION-1</p> <p>Old description:</p> <p>Client %COMMAND NAME% request message is a well-formed SOAP request (refer to onvif.xsd) AND</p> <p>Client %COMMAND NAME% request message has a proper hierarchy (refer to %SERVICE%.wsdl) AND</p> <p>New description:</p> <p>Client %COMMAND NAME% request messages are valid according to XML Schemas listed in Namespaces AND</p> <p>Client %COMMAND NAME% request in Test Procedure fulfills the following requirements:</p> <p>The following steps was removed because the requirements are fullfield by XML Schemas validation:</p> <ul style="list-style-type: none"> • MEDIASTREAMING-2: <ul style="list-style-type: none"> [S2] "<GetStreamUri>" includes tag: "<StreamSetup>" AND [S3] "<StreamSetup>" includes tag: "<Stream>" with ("RTP-Unicast" OR "RTP-Multicast") value AND [S4] "<StreamSetup>" includes tag: "<Transport>" AND • PTZCONTINUOUSPOSITIONING-1: <ul style="list-style-type: none"> [S3] "<ContinuousMove>" includes tag: "<Velocity>" AND [S5] "<PanTilt>" tag contains attribute: "x=" with value (example: -1, 0.1, 1, ...) AND [S6] "<PanTilt>" tag contains attribute: "y=" with value (example: -1, 0.1, 1, ...) AND • PTZCONTINUOUSPOSITIONING-2: <ul style="list-style-type: none"> [S3] "<ContinuousMove>" includes tag: "<Velocity>" AND [S5] "<Zoom>" tag contains attribute: "x=" with value (example: -1, 0.1, 1, ...) AND

		<ul style="list-style-type: none"> PTZABSOLUTEPOSITIONING-1: <ul style="list-style-type: none"> [S3] "<AbsoluteMove>" includes tag: "<Position>" AND [S5] "<PanTilt>" tag contains attribute: "x=" with value (example: -1, 0.1, 1, ...) AND [S6] "<PanTilt>" tag contains attribute: "y=" with value (example: -1, 0.1, 1, ...) AND PTZABSOLUTEPOSITIONING-2: <ul style="list-style-type: none"> [S3] "<AbsoluteMove>" includes tag: "<Position>" AND [S5] "<Zoom>" tag contains attribute: "x=" with value (example: -1, 0.1, 1, ...) AND PTZRELATIVEPOSITIONING-1: <ul style="list-style-type: none"> [S3] "<RelativeMove>" includes tag: "<Translation>" AND [S5] "<PanTilt>" tag contains attribute: "x=" with value (example: -1, 0.1, 1, ...) AND [S6] "<PanTilt>" tag contains attribute: "y=" with value (example: -1, 0.1, 1, ...) AND PTZRELATIVEPOSITIONING-2: <ul style="list-style-type: none"> [S3] "<RelativeMove>" includes tag: "<Translation>" AND [S5] "<Zoom>" tag contains attribute: "x=" with value (example: -1, 0.1, 1, ...) AND
16.07	Jan 11, 2016	<p>The following test cases were updated to check of corresponding between RTSP session and GetStreamUri: MULTICAST STREAMING USING RTSP</p> <p>Normative references were updated.</p>
16.07	Dec 30, 2015	<p>The following test cases were updated to check of corresponding between RTSP session and GetStreamUri: STREAMING OVER RTSP STREAMING OVER UDP STREAMING OVER HTTP</p> <p>Normative references were updated.</p>
16.01	Dec 28, 2015	<p>The following test cases were updated to check of media type in RTSP SETUP requests and to check of corresponding between RTSP session and GetStreamUri: MJPEG VIDEO STREAMING MPEG4 VIDEO STREAMING H264 VIDEO STREAMING G.711 AUDIO STREAMING G.726 AUDIO STREAMING AAC AUDIO STREAMING</p> <p>Normative references were updated.</p>
16.01	December 02, 2015	<ul style="list-style-type: none"> Media Streaming Feature was updated to require supporting of RTP/UDP or RTP/RTSP/HTTP/TCP.
16.07	Nov 27, 2015	<p>General item (Test Overview) was added</p> <p>Minor updates in formatting, typos and terms</p> <p>Metadata Configurations test cases and related feature were updated according review results.</p>
16.01	Sep 23, 2015	<p>Added new Test Cases sections: Metadata Configurations.</p> <p>PTZ SEND AUXILIARY COMMAND test case was updated</p>

15.06	Jun 10, 2015	No major changes were made, just minor formatting fixes.
15.05	May 20, 2015	No major changes were made, just minor grammatical corrections.
15.02	Feb 19, 2015	Pass criteria in VIDEOSTREAMING-1, 2 and 3 test cases have been updated (added check for Media Type: "video" in RTSP DESCRIBE response).
14.12	Dec 11, 2014	Fixed typos and inconsistencies.
1.2	Oct 29, 2014	<p>Changes were made in "PASS" criteria of the "5.4. STREAMING OVER RTSP", "6.2. MJPEG VIDEO STREAMING", "6.3. MPEG4 VIDEO STREAMING", "6.4. H264 VIDEO STREAMING" and "7.2. MULTICAST STREAMING USING RTSP" Test Cases.</p> <p>Test "5.4. STREAMING OVER RTSP" was divided into three different tests (RTSP, UDP and HTTP).</p> <p>New Test Case "8.3. GET SPECIFIC VIDEO ENCODER CONFIGURATION" has been added.</p> <p>Section "10.1. Expected Scenarios Under Test" has been updated.</p> <p>New Test Case "10.3. GET SPECIFIC VIDEO SOURCE CONFIGURATION" has been added.</p> <p>"ISO/IEC Directives, Part 2" reference has been added to "2. Normative references" section.</p> <p>The new section "3.1 Conventions" has been added.</p> <p>Specific Namespace prefixes have been removed from "PASS" criteria of all Test Cases.</p> <p>Fixed typos and inconsistencies.</p>
1.1	Sep 04, 2014	<p>MEDIASTREAMING-1, MEDIASTREAMING-2 and MEDIASTREAMING-3 test cases have been updated.</p> <p>Video Streaming Test Cases have been added.</p> <p>Multicast Streaming Test Cases have been added.</p> <p>Test Cases for Video Encoder Configurations have been added.</p> <p>Media Profile Configurations Test Cases have been added.</p> <p>Video Source Configurations Test Cases have been added.</p> <p>"Scope", "Security", "Capabilities" and "Event Handling" sections have been updated.</p>
1.0	Jul 31, 2014	Initial version. The first release includes MEDIASTREAMING-1 GET PROFILES, MEDIASTREAMING-2 GET STREAM URI and MEDIASTREAMING-3 STREAMING OVER RTSP test cases.

Table of Contents

1 Introduction 24

1.1 Scope 24

1.2 Test Cases for Profile Mandatory Features 24

1.2.1 Username Token 25

1.2.2 HTTP Digest 25

1.2.3 Capabilities 25

1.2.4 Media Streaming 25

1.2.5 Video Streaming 25

1.2.6 Video Encoder Configuration 25

1.2.7 Multiple Video Sources 25

1.3 Test Cases for Profile Conditional Features 25

1.3.1 Event Handling 25

1.3.2 Keep Alive for Pull Point Event Handling 26

1.3.3 Discovery 26

1.3.4 Network Video Transmitter Discovery Type Filter Test Cases 26

1.3.5 Network Configuration 26

1.3.6 System 27

1.3.7 User Handling 27

1.3.8 Relay Outputs 27

1.3.9 NTP 27

1.3.10 Dynamic DNS 27

1.3.11 Zero Configuration 27

1.3.12 IP Address Filtering 27

1.3.13 Multicast Streaming 27

1.3.14 Media Profile Configurations 27

1.3.15 Video Source Configuration 28

1.3.16 Audio Streaming 28

1.3.17 Metadata Configuration 28

1.3.18 Multiple Audio Sources 28

1.3.19 PTZ - Listing 28

1.3.20	PTZ - Configuration	28
1.3.21	PTZ Pan Tilt Continuous Positioning	28
1.3.22	PTZ Zoom Continuous Positioning	28
1.3.23	PTZ Pan Tilt Absolute Positioning	29
1.3.24	PTZ - Listing	29
1.3.25	PTZ Pan Tilt Relative Positioning	29
1.3.26	PTZ Zoom Relative Positioning	29
1.3.27	PTZ Presets	29
1.3.28	PTZ Home Position	29
1.3.29	PTZ - Auxiliary Command	29
1.4	Test Cases for Profile Optional Features	29
1.4.1	Set Synchronization Point (Event Service)	29
1.4.2	Unsubscribe	30
1.4.3	System Date and Time Configuration	30
1.4.4	Hostname Configuration	30
1.4.5	DNS Configuration	30
1.4.6	Network Protocols Configuration	30
1.5	Supplementary Features and Test Cases	30
2	Normative references	31
3	Terms and Definitions	33
3.1	Conventions	33
3.2	Definitions	33
3.3	Abbreviations	33
3.4	Namespaces	34
4	Test Overview	36
4.1	General	36
4.1.1	Feature Level Requirement	36
4.1.2	Expected Scenarios Under Test	36
4.1.3	Test Cases	37
4.2	Test Setup	37
4.3	Prerequisites	37

5	Test Cases for Profile Mandatory Features	39
5.1	Username Token Test Cases	39
5.1.1	Feature Level Requirement:	39
5.1.2	Expected Scenarios Under Test:	39
5.1.3	USER TOKEN PROFILE	39
5.2	HTTP Digest Test Cases	41
5.2.1	Feature Level Requirement:	41
5.2.2	Expected Scenarios Under Test:	41
5.2.3	HTTP DIGEST	42
5.3	Capabilities Test Cases	43
5.3.1	Feature Level Requirement:	43
5.3.2	Expected Scenarios Under Test:	43
5.3.3	GET SERVICES	44
5.3.4	GET CAPABILITIES	44
5.4	Media Streaming Test Cases	45
5.4.1	Feature Level Requirement:	45
5.4.2	Expected Scenarios Under Test:	46
5.4.3	GET PROFILES	46
5.4.4	GET STREAM URI	47
5.4.5	STREAMING OVER RTSP	48
5.4.6	STREAMING OVER UDP	51
5.4.7	STREAMING OVER HTTP	54
5.5	Video Streaming Test Cases	57
5.5.1	Feature Level Requirement:	57
5.5.2	Expected Scenarios Under Test:	57
5.5.3	MJPEG VIDEO STREAMING	57
5.5.4	MPEG4 VIDEO STREAMING	60
5.5.5	H264 VIDEO STREAMING	63
5.6	Video Encoder Configurations Test Cases	66
5.6.1	Feature Level Requirement:	66
5.6.2	Expected Scenarios Under Test:	66

5.6.3	LIST VIDEO ENCODER CONFIGURATIONS	67
5.6.4	GET SPECIFIC VIDEO ENCODER CONFIGURATION	68
5.6.5	GET VIDEO ENCODER CONFIGURATION OPTIONS	69
5.6.6	SET VIDEO ENCODER CONFIGURATION	70
5.7	Multiple Video Sources Test Cases	71
5.7.1	Feature Level Requirement:	71
5.7.2	Expected Scenarios Under Test:	71
5.7.3	STREAMING WITH ALL VIDEO SOURCES DETECTED IN GET PROFILES	72
6	Test Cases for Profile Conditional Features	75
6.1	Event Handling Test Cases	75
6.1.1	Feature Level Requirement:	75
6.1.2	Expected Scenarios Under Test:	75
6.1.3	PULLPOINT	76
6.1.4	BASE NOTIFICATION	77
6.1.5	METADATA STREAMING USING MEDIA	78
6.2	Keep Alive for Pull Point Event Handling Test Cases	80
6.2.1	Feature Level Requirement:	80
6.2.2	Expected Scenarios Under Test:	81
6.2.3	PULLPOINT	81
6.2.4	RENEW	83
6.2.5	PULL MESSAGES AS KEEP ALIVE	84
6.3	Discovery Test Cases	85
6.3.1	Feature Level Requirement:	85
6.3.2	Expected Scenarios Under Test:	85
6.3.3	WS-DISCOVERY	86
6.4	Network Video Transmitter Discovery Type Filter Test Cases	87
6.4.1	Feature Level Requirement:	87
6.4.2	Expected Scenarios Under Test:	87
6.4.3	NVT DISCOVERY TYPE FILTER	88
6.5	Network Configuration Test Cases	89

6.5.1	Feature Level Requirement:	89
6.5.2	Expected Scenarios Under Test:	90
6.5.3	GET NETWORK INTERFACES	90
6.5.4	SET NETWORK INTERFACES	91
6.5.5	GET NETWORK DEFAULT GATEWAY	92
6.5.6	SET NETWORK DEFAULT GATEWAY	93
6.6	System Test Cases	94
6.6.1	Feature Level Requirement:	94
6.6.2	Expected Scenarios Under Test:	95
6.6.3	GET DEVICE INFORMATION	95
6.7	User Handling Test Cases	96
6.7.1	Feature Level Requirement:	96
6.7.2	Expected Scenarios Under Test:	96
6.7.3	CREATE USERS	97
6.7.4	GET USERS	98
6.7.5	SET USER	99
6.7.6	DELETE USERS	100
6.8	Relay Outputs Test Cases	101
6.8.1	Feature Level Requirement:	101
6.8.2	Expected Scenarios Under Test:	101
6.8.3	GET RELAY OUTPUTS	102
6.8.4	SET RELAY OUTPUT STATE	103
6.8.5	SET RELAY OUTPUT SETTINGS BISTABLE MODE	104
6.8.6	SET RELAY OUTPUT SETTINGS MONOSTABLE MODE	105
6.9	NTP Test Cases	106
6.9.1	Feature Level Requirement:	106
6.9.2	Expected Scenarios Under Test:	107
6.9.3	GET NTP	107
6.9.4	SET NTP	108
6.10	Dynamic DNS Test Cases	109
6.10.1	Feature Level Requirement:	109

6.10.2	Expected Scenarios Under Test:	109
6.10.3	GET DYNAMIC DNS SETTINGS	109
6.10.4	SET DYNAMIC DNS SETTINGS	110
6.11	Zero Configuration Test Cases	111
6.11.1	Feature Level Requirement:	111
6.11.2	Expected Scenarios Under Test:	111
6.11.3	GET ZERO CONFIGURATION	112
6.11.4	SET ZERO CONFIGURATION	113
6.12	IP Address Filtering Test Cases	114
6.12.1	Feature Level Requirement:	114
6.12.2	Expected Scenarios Under Test:	114
6.12.3	GET IP ADDRESS FILTER	115
6.12.4	SET IPv4 ADDRESS FILTER	116
6.12.5	SET IPv6 ADDRESS FILTER	117
6.12.6	ADD IPv4 ADDRESS FILTER	118
6.12.7	ADD IPv6 ADDRESS FILTER	119
6.12.8	REMOVE IPv4 ADDRESS FILTER	120
6.12.9	REMOVE IPv6 ADDRESS FILTER	121
6.13	Multicast Streaming Test Cases	122
6.13.1	Feature Level Requirement:	122
6.13.2	Expected Scenarios Under Test:	122
6.13.3	MULTICAST STREAMING USING RTSP	123
6.13.4	MULTICAST STREAMING USING SOAP	126
6.14	Media Profile Configurations Test Cases	127
6.14.1	Feature Level Requirement:	127
6.14.2	Expected Scenarios Under Test:	127
6.14.3	LIST AVAILABLE MEDIA PROFILES	128
6.14.4	GET SPECIFIC MEDIA PROFILE	129
6.14.5	CREATE A MEDIA PROFILE	130
6.15	Video Source Configurations Test Cases	131
6.15.1	Feature Level Requirement:	131

- 6.15.2 Expected Scenarios Under Test: 131
- 6.15.3 LIST VIDEO SOURCE CONFIGURATIONS 132
- 6.15.4 GET SPECIFIC VIDEO SOURCE CONFIGURATION 133
- 6.15.5 GET VIDEO SOURCE CONFIGURATION OPTIONS 134
- 6.15.6 SET VIDEO SOURCE CONFIGURATION 135
- 6.15.7 GET COMPATIBLE VIDEO SOURCE CONFIGURATIONS 136
- 6.15.8 ADD VIDEO SOURCE CONFIGURATION 137
- 6.16 Audio Streaming Test Cases 139
 - 6.16.1 Feature Level Requirement: 139
 - 6.16.2 Expected Scenarios Under Test: 139
 - 6.16.3 CONFIGURE MEDIA PROFILE FOR AUDIO STREAMING 140
 - 6.16.4 G.711 AUDIO STREAMING 142
 - 6.16.5 G.726 AUDIO STREAMING 145
 - 6.16.6 AAC AUDIO STREAMING 148
- 6.17 Metadata Configurations Test Cases 151
 - 6.17.1 Feature Level Requirement: 151
 - 6.17.2 Expected Scenarios Under Test: 151
 - 6.17.3 LIST METADATA CONFIGURATIONS 152
 - 6.17.4 GET SPECIFIC METADATA CONFIGURATION 153
 - 6.17.5 GET METADATA CONFIGURATION OPTIONS 154
 - 6.17.6 MODIFY METADATA CONFIGURATION 155
- 6.18 Multiple Audio Sources Test Cases 156
 - 6.18.1 Feature Level Requirement: 156
 - 6.18.2 Expected Scenarios Under Test: 156
 - 6.18.3 STREAMING WITH ALL AUDIO SOURCES DETECTED IN GET
PROFILES 157
- 6.19 PTZ - Listing Test Cases 159
 - 6.19.1 Feature Level Requirement: 159
 - 6.19.2 Expected Scenarios Under Test: 159
 - 6.19.3 GET NODES 159
 - 6.19.4 GET NODE 160

6.20	PTZ - Configuration Test Cases	161
6.20.1	Feature Level Requirement:	161
6.20.2	Expected Scenarios Under Test:	161
6.20.3	ADD PTZ CONFIGURATION	162
6.21	PTZ Pan Tilt Continuous Positioning Test Cases	163
6.21.1	Feature Level Requirement:	163
6.21.2	Expected Scenarios Under Test:	163
6.21.3	PTZ CONTINUOUS MOVE PAN/TILT	164
6.21.4	PTZ PAN TILT STOP	165
6.21.5	STOP PAN TILT MOVEMENT USING PTZ CONTINUOUS MOVE	166
6.22	PTZ Zoom Continuous Positioning Test Cases	167
6.22.1	Feature Level Requirement:	167
6.22.2	Expected Scenarios Under Test:	168
6.22.3	PTZ CONTINUOUS MOVE ZOOM	168
6.22.4	PTZ ZOOM STOP	169
6.22.5	STOP ZOOM MOVEMENT USING PTZ CONTINUOUS MOVE	170
6.23	PTZ Pan Tilt Absolute Positioning Test Cases	172
6.23.1	Feature Level Requirement:	172
6.23.2	Expected Scenarios Under Test:	172
6.23.3	PTZ ABSOLUTE MOVE PAN/TILT	172
6.24	PTZ Zoom Absolute Positioning Test Cases	173
6.24.1	Feature Level Requirement:	173
6.24.2	Expected Scenarios Under Test:	174
6.24.3	PTZ ABSOLUTE MOVE ZOOM	174
6.25	PTZ Pan Tilt Relative Positioning Test Cases	175
6.25.1	Feature Level Requirement:	175
6.25.2	Expected Scenarios Under Test:	175
6.25.3	PTZ RELATIVE MOVE PAN/TILT	176
6.26	PTZ Zoom Relative Positioning Test Cases	177
6.26.1	Feature Level Requirement:	177
6.26.2	Expected Scenarios Under Test:	177

6.26.3	PTZ RELATIVE MOVE ZOOM	177
6.27	PTZ Presets Test Cases	178
6.27.1	Feature Level Requirement:	178
6.27.2	Expected Scenarios Under Test:	179
6.27.3	PTZ GET PRESETS	179
6.27.4	PTZ GOTO PRESET	180
6.28	PTZ Home Position Test Cases	181
6.28.1	Feature Level Requirement:	181
6.28.2	Expected Scenarios Under Test:	181
6.28.3	PTZ HOME POSITION	181
6.29	PTZ - Auxiliary Command Test Cases	182
6.29.1	Feature Level Requirement:	182
6.29.2	Expected Scenarios Under Test:	183
6.29.3	PTZ SEND AUXILIARY COMMAND	183
7	Test Cases for Profile Optional Features	185
7.1	Set Synchronization Point (Event Service) Test Cases	185
7.1.1	Feature Level Requirement:	185
7.1.2	Expected Scenarios Under Test:	185
7.1.3	SET SYNCHRONIZATION POINT (EVENT SERVICE)	185
7.2	Unsubscribe Test Cases	186
7.2.1	Expected Scenarios Under Test:	187
7.2.2	UNSUBSCRIBE	187
7.3	System Date and Time Configuration Test Cases	188
7.3.1	Feature Level Requirement:	188
7.3.2	Expected Scenarios Under Test:	188
7.3.3	GET SYSTEM DATE AND TIME	189
7.3.4	SET SYSTEM DATE AND TIME	190
7.4	Hostname Configuration Test Cases	191
7.4.1	Feature Level Requirement:	191
7.4.2	Expected Scenarios Under Test:	191
7.4.3	GET HOSTNAME	192

7.4.4	SET HOSTNAME	193
7.5	DNS Configuration Test Cases	194
7.5.1	Feature Level Requirement:	194
7.5.2	Expected Scenarios Under Test:	194
7.5.3	GET DNS	194
7.5.4	SET DNS	195
7.6	Network Protocols Configuration Test Cases	196
7.6.1	Feature Level Requirement:	196
7.6.2	Expected Scenarios Under Test:	196
7.6.3	GET NETWORK PROTOCOLS	197
7.6.4	SET NETWORK PROTOCOLS	198
8	Supplementary Features and Test Cases	200
8.1	METADATA STREAMING USING MEDIA2	200
A	Test for Appendix A	203
A.1	Get Video Sources List from GetProfiles responses	203
A.2	Get Video Source Token That was Used for Streaming	203
A.3	Find Video Streaming corresponding to GetStreamUri	205
A.4	Get Audio Sources List from GetProfiles responses	207
A.5	Get Audio Source Token That was Used for Streaming	208
A.6	Find Audio Streaming corresponding to GetStreamUri	210
A.7	Required Number of Devices Summary	212

1 Introduction

The goal of the ONVIF Test Specification set is to make it possible to realize fully interoperable IP physical security implementations from different vendors. This specification also acts as an input document to the development of a test tool which will be used to test the ONVIF Client implementation conformance towards ONVIF standard. This Client Test Tool analyzes network communications between ONVIF Devices and Clients being tested and determines whether a specific Client is ONVIF conformant (see ONVIF Conformance Process Specification).

This particular document defines test cases required for testing Profile S features of a Client application e.g. Media Streaming, Video Streaming, Multicast Streaming, Video Encoder Configuration, Media Profile Creation and Video Source Configuration. It also describes the test framework, test setup, prerequisites, test policies needed for the execution of the described test cases.

1.1 Scope

This ONVIF Profile S Client Test Specification defines and regulates the conformance testing procedure for the ONVIF conformant Clients in the scope of Profile S features. Conformance testing is meant to be black-box network traces analysis and verification. The objective of this specification is to provide the test cases to test individual requirements of ONVIF Clients in the scope of Profile S features according to ONVIF Profile Specifications.

The principal intended purposes are:

- Provide self-assessment tool for implementations.
- Provide comprehensive test suite coverage for Profile S features.

This specification **does not** address the following:

- Product use cases and non-functional (performance and regression) testing and analysis.
- SOAP Implementation Interoperability test i.e. Web Services Interoperability Basic Profile version 2.0 (WS-I BP2.0).
- Network protocol implementation Conformance test for HTTPS, HTTP, RTP and RTSP protocols.

The following sections cover test cases needed for the verification of relevant features as mentioned in the ONVIF Profile Specifications.

1.2 Test Cases for Profile Mandatory Features

This section defines test cases which are mandatory for Profile S Client conformance.

1.2.1 Username Token

Username Token section defines security mechanism for Username Token Profile.

1.2.2 HTTP Digest

HTTP Digest section defines security mechanism for HTTP Digest Authentication.

1.2.3 Capabilities

Capabilities section specifies Client ability to retrieve available services and advanced functionalities which are offered by a Device.

1.2.4 Media Streaming

Media Streaming section defines different streaming options based on RTP protocol which are required for all types of streams of video, audio and metadata. Media control is done using RTSP protocol.

1.2.5 Video Streaming

Video Streaming section specifies Client ability to establish specific video streams in MJPEG, MPEG4 and H264 video formats.

1.2.6 Video Encoder Configuration

Video Encoder Configurations section specifies listing and modification of video encoder configurations on Device.

1.2.7 Multiple Video Sources

Multiple Video Sources section specifies Client ability to initiate video streaming for all Video Sources returned by Device in GetProfilesResponse.

1.3 Test Cases for Profile Conditional Features

This section defines test cases which are mandatory for Profile S Client conformance.

1.3.1 Event Handling

Event Handling section defines Client ability to initiate and receive notifications (events) from a Device.

The event handling test cases cover the following mandatory interfaces:

- Pull Point Notification Interface
 - This test specification provides test cases to verify the implementation of the PullPoint Interface of a Client.
- Basic Notification Interface
 - This test specification provides test cases to verify the implementation of the Basic Notification Interface of a Client.
- Metadata Streaming Interface
 - This test specification provides test cases to verify the implementation of the Metadata Streaming Interface of a Client using Media Service and using Media2 Service.

1.3.2 Keep Alive for Pull Point Event Handling

Keep Alive for Pull Point Event Handling section specifies Client ability to use keep alive for Pull Point Event Handling using PullMessages or Renew approach.

1.3.3 Discovery

Discovery section defines Client ability to locate services on a local network using Web Services Dynamic Discovery (WS-Discovery) protocol. It uses IP multicast address 239.255.255.250 and TCP and UDP port 3702 and SOAP-over-UDP standard for communication between nodes.

1.3.4 Network Video Transmitter Discovery Type Filter Test Cases

Network Video Transmitter Discovery Type Filter Test Cases section defines Client ability to locate services, which are support Network Video Transmitter Discovery Type on a local network using Web Services Dynamic Discovery (WS-Discovery) protocol. It uses IPv4 address 239.255.255.250 or multicast IPv6 address [FF02::C] and port 3702 with Types filter that contains dn:NetworkVideoTransmitter or with skipped Types filter.

1.3.5 Network Configuration

Network Configuration section defines Client ability to obtain and configure of network settings on Device.

1.3.6 System

System section defines Client ability to obtain Device information and configure of system settings on Device.

1.3.7 User Handling

User Handling section defines Client ability to manage users on Device.

1.3.8 Relay Outputs

Relay Outputs section defines Client ability to list, configure and trigger relay outputs on Device.

1.3.9 NTP

NTP section defines Client ability to configure synchronization of time using NTP servers on Device.

1.3.10 Dynamic DNS

Dynamic DNS section defines Client ability to configure dynamic DNS settings on Device.

1.3.11 Zero Configuration

Zero Configuration section defines Client ability to enable or disable zero configuration on Device.

1.3.12 IP Address Filtering

IP Address Filtering section defines Client ability to manage IP address filters on Device.

1.3.13 Multicast Streaming

Multicast Streaming section specifies Client ability to initiate multicast stream by using StartMulticastStreaming and StopMulticastStreaming operations or by using RTSP SETUP command with multicast transport parameter.

1.3.14 Media Profile Configurations

Media Profile Configurations section specifies creation and retrieval of Media Profile Configurations from Device.

1.3.15 Video Source Configuration

Video Source Configurations section specifies listing and modification of video source configurations on Device.

1.3.16 Audio Streaming

Audio Streaming section specifies Client ability to initiate audio stream in G.711, G.726 and AAC encoding formats. This section also specifies Client ability to configure a media profile for audio streaming.

1.3.17 Metadata Configuration

Metadata Configurations section specifies listing and modification of metadata configurations on Device.

1.3.18 Multiple Audio Sources

Multiple Audio Sources section specifies Client ability to initiate audio streaming for all Audio Sources returned by Device in GetProfilesResponse.

1.3.19 PTZ - Listing

PTZ - Listing section specifies Client ability to read PTZ capabilities.

1.3.20 PTZ - Configuration

PTZ - Configuration section specifies Client ability to add PTZ configuration to a media profile.

1.3.21 PTZ Pan Tilt Continuous Positioning

PTZ Pan Tilt Continuous Move section specifies Client ability to move a PTZ Device using ContinuousMove operation for Pan Tilt and stop ongoing pan tilt movement using Stop operation or sending zero values for Pan/Tilt.

1.3.22 PTZ Zoom Continuous Positioning

PTZ Zoom Continuous Move section specifies Client ability to move a PTZ Device using ContinuousMove operation for Zoom and stop ongoing pan tilt movement using Stop operation or sending zero values for Zoom.

1.3.23 PTZ Pan Tilt Absolute Positioning

PTZ Pan Tilt Absolute Positioning section specifies Client ability to move a PTZ Device using the AbsoluteMove operation for Pan Tilt.

1.3.24 PTZ - Listing

PTZ - Listing section specifies Client ability to read PTZ capabilities.

1.3.25 PTZ Pan Tilt Relative Positioning

PTZ Pan Tilt Relative Positioning section specifies Client ability to move a PTZ Device using the RelativeMove operation for Pan Tilt.

1.3.26 PTZ Zoom Relative Positioning

PTZ Zoom Relative Positioning section specifies Client ability to move a PTZ Device using the RelativeMove operation for Zoom.

1.3.27 PTZ Presets

PTZ Presets section specifies Client ability to list the presets of a PTZ Node and move a PTZ Device to a specific preset.

1.3.28 PTZ Home Position

PTZ Home Position section specifies Client ability to move a PTZ Device to its home position.

1.3.29 PTZ - Auxiliary Command

PTZ - Auxiliary Command section specifies Client ability to send auxiliary commands to a PTZ Device.

1.4 Test Cases for Profile Optional Features

This section defines test cases which are optional for Profile S Client conformance.

1.4.1 Set Synchronization Point (Event Service)

Set Synchronization Point section defines Client ability to synchronize its properties with the properties of the device using SetSynchronizationPoint operation.

1.4.2 Unsubscribe

Unsubscribe section defines Client ability to terminate subscription using Unsubscribe operation.

1.4.3 System Date and Time Configuration

System Date and Time Configuration section defines Client ability to configure Device system date and time using GetSystemDateAndTime and SetSystemDateAndTime operations.

1.4.4 Hostname Configuration

Hostname Configuration section defines Client ability to obtain and configure of hostname settings on Device.

1.4.5 DNS Configuration

DNS Configuration section defines Client ability to obtain and configure of DNS settings on Device.

1.4.6 Network Protocols Configuration

Network Protocols Configuration section defines Client ability to obtain and configure of network protocols settings on Device.

1.5 Supplementary Features and Test Cases

This section defines supplementary features and test cases which are not the part of profile, but Profile S Features results depends on them.

2 Normative references

- ONVIF Conformance Process Specification:
<https://www.onvif.org/profiles/conformance/>
- ONVIF Profile Policy:
<https://www.onvif.org/profiles/>
- ONVIF Network Interface Specifications:
<https://www.onvif.org/profiles/specifications/>
- ISO/IEC Directives, Part 2, Annex H:
www.iso.org/directives
- ISO 16484-5:2014-09 Annex P:
<https://www.iso.org/obp/ui/#!iso:std:63753:en>
- WS-BaseNotification:
http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf
- W3C SOAP 1.2, Part 1, Messaging Framework:
<http://www.w3.org/TR/soap12-part1/>
- W3C XML Schema Part 1: Structures Second Edition:
<http://www.w3.org/TR/xmlschema-1/>
- W3C XML Schema Part 2: Datatypes Second Edition:
["http://www.w3.org/TR/xmlschema-2/](http://www.w3.org/TR/xmlschema-2/) [<http://www.w3.org/TR/xmlschema-2/>]
- W3C Web Services Addressing 1.0 – Core:
<http://www.w3.org/TR/ws-addr-core/>
- ONVIF Profile S Specification:
<https://www.onvif.org/profiles/profile-s/>
- IETF RFC 2435, RTP Payload Format for JPEG-compressed Video:
<http://www.ietf.org/rfc/rfc2435.txt>

- IETF RFC 3016, RTP Payload Format for MPEG-4 Audio/Visual Streams:
<http://www.ietf.org/rfc/rfc3016>
- IETF RFC 3984, RTP Payload Format for H.264 Video:
<http://www.ietf.org/rfc/rfc3984>
- IETF RFC 3640, RTP Payload Format for Transport of MPEG-4 Elementary Streams:
<http://www.ietf.org/rfc/rfc3640>
- IETF RFC 2326, Real Time Streaming Protocol (RTSP):
<http://www.ietf.org/rfc/rfc2326.txt>

3 Terms and Definitions

3.1 Conventions

The key words "shall", "shall not", "should", "should not", "may", "need not", "can", "cannot" in this specification are to be interpreted as described in [ISO/IEC Directives Part 2].

3.2 Definitions

This section describes terms and definitions used in this document.

Address	An address refers to a URI.
Profile	See ONVIF Profile Policy.
ONVIF Device	Computer appliance or software program that exposes one or multiple ONVIF Web Services.
ONVIF Client	Computer appliance or software program that uses ONVIF Web Services.
Conversation	A Conversation is all exchanges between two MAC addresses that contains SOAP request and response.
Network	A network is an interconnected group of devices communicating using the Internet protocol.
Network Trace Capture file	Data file created by a network protocol analyzer software (such as Wireshark). Contains network packets data recorded during a live network communications.
SOAP	SOAP is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. It uses XML technologies to define an extensible messaging framework providing a message construct that can be exchanged over a variety of underlying protocols.
Client Test Tool	ONVIF Client Test Tool that tests ONVIF Client implementation towards the ONVIF Test Specification set.
Valid Device Response	Device has responded to specific request with code HTTP or RTSP 200 OK and SOAP fault message has not appeared.
Profile S	The Profile S Specification.
Media Profile	A media profile maps a video and/or audio source to a video and/or an audio encoder, PTZ and analytics configurations.

3.3 Abbreviations

This section describes abbreviations used in this document.

HTTP	Hyper Text Transport Protocol.
HTTPS	Hyper Text Transport Protocol over Secure Socket Layer.

IP	Internet Protocol.
IPv4	Internet Protocol version 4.
TCP	Transport Control Protocol.
UDP	User Datagram Protocol.
URI	Uniform Resource Identifier.
WSDL	Web Services Description Language.
XML	eXtensible Markup Language.
JPEG	Joint Photographic Expert Group.
MPEG-4	Moving Picture Experts Group 4.
RTP	Real-time Transport Protocol.
RTSP	Real Time Streaming Protocol.
SDP	Session Description Protocol.

3.4 Namespaces

Prefix and namespaces used in this test specification are listed in Table 1. These prefixes are not part of the standard and an implementation can use any prefix.

Table 3.1. Defined namespaces in this specification

Prefix	Namespace URI	Description
soapenv	http://www.w3.org/2003/05/soap-envelope	Envelope namespace as defined by SOAP 1.2 [SOAP 1.2, Part 1]
xs	http://www.w3.org/2001/XMLSchema	Instance namespace as defined by XS [XML-Schema, Part1] and [XMLSchema,Part 2]
xsi	http://www.w3.org/2001/XMLSchema-instance	XML schema instance namespace
tns1	http://www.onvif.org/ver10/topics	The namespace for the ONVIF topic namespace
tt	http://www.onvif.org/ver10/schema	ONVIF XML schema descriptions
tds	http://www.onvif.org/ver10/device/wsdl	The namespace for the WSDL device service
tev	http://www.onvif.org/ver10/events/wsdl	The namespace for the WSDL event service
ter	http://www.onvif.org/ver10/error	The namespace for ONVIF defined faults
wsnt	http://docs.oasis-open.org/wsn/b-2	Schema namespace of the [WS-BaseNotification] specification.
wsa	http://www.w3.org/2005/08/addressing	Device addressing namespace as defined by [WS-Addressing].

Prefix	Namespace URI	Description
trt	http://www.onvif.org/ver10/media/wsdl	The namespace for the WSDL media service

4 Test Overview

This section provides information for the test setup procedure and required prerequisites that should be followed during test case execution.

An ONVIF Client compliant to the Profile S is an ONVIF Client that can configure, request, and control streaming of video data over an IP network from an ONVIF Device compliant to the Profile S. The Profile S also includes receiving Audio and Metadata Stream, and Relay Outputs.

An ONVIF Profile is described by a fixed set of functionalities through a number of services that are provided by the ONVIF standard. A number of services and functionalities are mandatory for each type of ONVIF Profile. An ONVIF Device and ONVIF Client may support any combination of Profiles and other optional services and functionalities.

4.1 General

Test Cases are grouped depending on features. Each Test Cases group provides description of feature requirement level for Profiles, expected scenario under test and related test cases:

- Feature Level Requirement
- Expected Scenarios Under Test
- List of Test Cases

4.1.1 Feature Level Requirement

Feature Level Requirement item contains a feature ID, check condition based on Device features, required number of Devices and feature requirement level for the Profiles, which will be used for Profiles conformance.

To claim this Feature as supported Client shall pass Expected Scenario Under Test:

- for each Device, which supports Device Features defined in Check Condition Based on Device Features
- for at least with number of Devices specified in Required Number of Devices

If Feature Level Requirement is defined as Mandatory for some Profile, Client shall support this Feature to claim this Profile Conformance.

4.1.2 Expected Scenarios Under Test

Expected Scenarios Under Test item contains expected scenario under test, conditions when the feature will be defined as supported and as not supported.

4.1.3 Test Cases

Test Case items contain list of test cases which are related to feature. Test cases provide exact procedure of testing feature support conditions.

Each Test Case contains the following parts:

- Test Label - Unique label for each test
- Test Case ID - Unique ID for each test
- Profile Normative References - Requirement level for the feature under test is defined in Profile Specification. This reference is informative and will not be used in conformance procedure.
- Feature Under Test - Feature which is under current test. Typically a particular command or an event.
- Test Purpose - The purpose of current test case.
- Pre-Requisite - The pre-requisite defines when the test should be performed. In case if pre-requisite does not match, the test result will be NOT DETECTED.
- Test Procedure - scenario expected to be reflected in network trace file.
- Test Result - Passed and failed criteria of the test case. Depending on these criteria test result will be defined as PASSED or FAILED.

4.2 Test Setup

Collect Network traces files required by the test cases.

Collect Feature List XML files for Devices detected in the Network Trace files.

Client shall support all mandatory and conditional features listed in the Device Feature List XML file supplied for the Profiles supported by the Client.

For compatibility with the Profile S, the ONVIF Client shall follow the requirements of the conformance process. For details please see the latest ONVIF Conformance Process Specification.

4.3 Prerequisites

The pre-requisites for executing the test cases described in this Test Specification include:

The Device shall be configured with an IPv4 address.

The Device shall be able to be discovered by the Client.

5 Test Cases for Profile Mandatory Features

5.1 Username Token Test Cases

5.1.1 Feature Level Requirement:

Validated Feature: Username Token Authentication (UserTokenProfile)

Check Condition based on Device Features: WS-Username Token

Required Number of Devices: 1 (Note: Username Token feature shall be passed with at least one Device and can be not detected with other devices with supporting of WS-Username Token)

Profile S Requirement: Mandatory

Profile A Requirement: None

Profile C Requirement: None

Profile G Requirement: None

Profile T Requirement: None

5.1.2 Expected Scenarios Under Test:

1. Client invokes a specific command which requires authentication with WS-Username Token authentication header.
2. Device sends a valid response to this request.
3. Client is considered as supporting WS-Username Token if the following conditions are met:
 - Device returns a valid response to specific request with UsernameToken authentication header.
4. Client is considered as NOT supporting WS-Username Token if the following is TRUE:
 - All UsernameToken attempts detected are failed.

5.1.3 USER TOKEN PROFILE

Test Label: Security - User token profile

Test Case ID: USERTOKENPROFILE-1

Feature	Under	Test:	Username	Token	Authentication
(UserTokenProfile_UsernameTokenAuthentication)					

Test Purpose: To verify that the Client supports the User Token Profile for Message level security.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with UsernameToken Authentication present.

Test Procedure (expected to be reflected in network trace file):

1. Client sends a request (e.g. GetUsers) to the Device with correctly formatted UsernameToken.
2. Verify that the Device accepts the correct request.

Test Result:**PASS -**

- Client request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client request that contains UsernameToken authentication in SOAP header fulfills the following requirements:
 - [S1] Client request contains "<Security>" tag after the "<Header>" tag AND
 - [S2] "<Security>" includes tag: "<UsernameToken>" AND
 - [S3] "<UsernameToken>" includes tag: "<Username>" AND
 - [S4] "<UsernameToken>" includes tag: "<Password>" AND
 - [S5] "<UsernameToken>" includes tag: "<Nonce>" AND
 - [S6] "<UsernameToken>" includes tag: "<Created>" AND
 - [S7] Device response contains "HTTP/* 200 OK" AND
 - [S8] Device response does NOT contain "<Fault>" tag.

FAIL -

- The Client failed PASS criteria.

5.2 HTTP Digest Test Cases

5.2.1 Feature Level Requirement:

Validated Feature: HTTP Digest authentication (HTTPDigest)

Check Condition based on Device Features: Digest

Required Number of Devices: 3

Profile A Requirement: Mandatory

Profile C Requirement: Mandatory

Profile D Requirement: Mandatory

Profile G Requirement: Mandatory

Profile S Requirement: Mandatory

Profile T Requirement: Mandatory

Profile M Requirement: Mandatory

5.2.2 Expected Scenarios Under Test:

1. Client invokes a specific command which is under testing without any user credentials (no UsernameToken, no HTTP Digest authentication header).
2. Device returns HTTP 401 Unauthorized error along with WWW-Authentication: Digest header.
3. Client re-sends request with HTTP Digest Authentication header corresponding to header provided in device response.
4. Device sends a valid response to this request.
5. Client is considered as supporting HTTP Digest if the following conditions are met:
 - Device returns a valid response to specific request with HTTP Digest authentication header.
6. Client is considered as NOT supporting HTTP Digest if the following is TRUE:
 - All HTTP Digest attempts detected are failed.

5.2.3 HTTP DIGEST

Test Label: Security - HTTP Digest Authentication.

Test Case ID: HTTPDIGEST-1

Feature Under Test: HTTP Digest (HTTPDigest_HTTPDigestAuthentication)

Test Purpose: To verify that the Client supports the HTTP Digest Authentication for HTTP level security.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with HTTP Digest Authentication present.

Test Procedure (expected to be reflected in network trace file):

1. Client sends a request that requires authentication (e.g. GetUsers) to the Device without any authentication.
2. Device rejects the request with HTTP error code 401 AND an HTTP Digest challenge.
3. Client sends a valid request with HTTP Digest Authentication.
4. Device accepts the correct request with response code HTTP 200 OK.

Test Result:

PASS -

- [S1] Client request contains (HTTP GET method OR HTTP POST method) without any authentication AND
- Client HTTP GET request has a proper hierarchy (refer to [RFC 1945]) AND
 - [S2] Device response contains "HTTP/* 401 Unauthorized" AND
 - [S3] Device response contains "realm=*" element AND
 - [S4] Device response contains "nonce=*" element AND
 - [S5] Client request contains (HTTP GET method OR HTTP POST method) with "Authorization: Digest username=*" element AND
- Client HTTP GET request with HTTP Authentication has a proper hierarchy (refer to [RFC 1945]) AND

- [S6] Client request contains "realm=" element with value from Device response AND
- [S7] Client request contains "nonce=" element with value from Device response AND
- [S8] Client request contains "uri=" element AND
- [S9] Device response contains "HTTP/* 200 OK".

FAIL -

- The Client failed PASS criteria.

5.3 Capabilities Test Cases

5.3.1 Feature Level Requirement:

Validated Feature: Capabilities (Capabilities)

Check Condition based on Device Features: None

Required Number of Devices: 3

Profile A Requirement: Mandatory

Profile C Requirement: Mandatory

Profile G Requirement: Mandatory

Profile S Requirement: Mandatory

Profile T Requirement: Mandatory

5.3.2 Expected Scenarios Under Test:

1. Client invokes a specific Capabilities command which is under testing.
2. Client is considered as supporting Capabilities if the following conditions are met:
 - Device returns a valid response to GetServices request OR
 - Device returns a valid response to GetCapabilities request.
3. Client is considered as NOT supporting Capabilities if the following is TRUE:
 - No Valid Device Response to GetServices request AND
 - No Valid Device Response to GetCapabilities request.

5.3.3 GET SERVICES

Test Label: Capabilities - Determine the available Services

Test Case ID: CAPABILITIES-1

Feature Under Test: Get Services (Capabilities_GetServicesRequest)

Test Purpose: To verify that Device Capabilities is received using GetServices request.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetServices command present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetServices request message to retrieve all services of the Device.
2. Verify that GetServicesResponse message from the Device contains code HTTP 200 OK without SOAP Fault.

Test Result:

PASS -

- Client **GetServices** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetServices** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetServices>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND
 - [S3] Device response contains "<GetServicesResponse>" tag.

FAIL -

- The Client failed PASS criteria.

5.3.4 GET CAPABILITIES

Test Label: Capabilities - Get Device Capabilities

Test Case ID: CAPABILITIES-2

Feature Under Test: Get Capabilities (Capabilities_GetCapabilities)

Test Purpose: To verify that Device Capabilities is received using GetCapabilities request.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetCapabilities command present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetCapabilities request message to retrieve Device Capabilities of the Device.
2. Verify that GetCapabilitiesResponse response message from the Device contains code HTTP 200 OK without SOAP Fault.

Test Result:

PASS -

- Client **GetCapabilities** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetCapabilities** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetCapabilities>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND
 - [S3] Device response contains "<GetCapabilitiesResponse>" tag.

FAIL -

- The Client failed PASS criteria.

5.4 Media Streaming Test Cases

5.4.1 Feature Level Requirement:

Validated Feature: Media Streaming (MediaStreaming)

Check Condition based on Device Features: Real Time Streaming (Media Service) is supported by Device.

Required Number of Devices: 3

Profile S Requirement: Mandatory

5.4.2 Expected Scenarios Under Test:

1. Client connects to Device to initiate Media Streaming.
2. Client is considered as supporting Media Streaming if the following conditions are met:
 - Device returns a valid response to GetProfiles request AND
 - Device returns a valid response to GetStreamURI request AND
 - Stream was successfully established by Client using UDP protocol OR HTTP protocol.
 - Stream was successfully established by Client using RTSP protocol (if supported).
3. Client is considered as NOT supporting Media Streaming if the following is TRUE:
 - No Valid Device Response to GetProfiles request OR
 - No Valid Device Response to GetStreamURI request OR
 - Client is unable to establish stream using UDP protocol OR HTTP protocol OR
 - Client is unable to establish stream using RTSP protocol if detected.

5.4.3 GET PROFILES

Test Label: Media Streaming - GetProfiles

Test Case ID: MEDIASTREAMING-1

Feature Under Test: Get Profiles (MediaStreaming_GetProfiles)

Test Purpose: To verify that list of media profiles from Device is received by Client using the GetProfiles operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetProfiles operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetProfiles request message to retrieve complete profiles list from Device.
2. Device responds with code HTTP 200 OK and GetProfilesResponse message.

Test Result:

PASS -

- Client **GetProfiles** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetProfiles** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:GetProfiles** AND
- Device response on the **GetProfiles** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **trt:GetProfilesResponse**.

FAIL -

- The Client failed PASS criteria.

5.4.4 GET STREAM URI

Test Label: Media Streaming - GetStreamURI

Test Case ID: MEDIASTREAMING-2

Feature Under Test: Get Stream URI (MediaStreaming_GetStreamURI)

Test Purpose: To verify that stream URI from Device is received by Client using the GetStreamURI operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetStreamURI operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetStreamUri request message with the Stream Setup element (contains two parts: Stream Type and Transport protocol) and Profile Token element (indicates the media profile selected).
2. Device responds with code HTTP 200 OK and GetStreamUriResponse message.

Test Result:**PASS -**

- Client **GetStreamUri** request messages are valid according to XML Schemas listed in [Namespaces](#) AND

- Client **GetStreamUri** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:GetStreamUri** AND
 - [S2] **trt:GetStreamUri\trt:StreamSetup\trt:Transport\trt:Protocol** element value is equal EITHER "UDP" OR "HTTP" OR "RTSP" AND
 - [S2] **trt:GetStreamUri\trt:ProfileToken** element has non-empty string value AND
- Device response on the **GetStreamUri** request fulfills the following requirements:
 - [S3] It has HTTP 200 response code AND
 - [S4] **soapenv:Body** element has child element **trt:GetStreamUriResponse**.

FAIL -

- The Client failed PASS criteria.

5.4.5 STREAMING OVER RTSP

Test Label: Media Streaming - RTSP

Test Case ID: MEDIASTREAMING-3

Feature Under Test: Streaming Over RTSP (MediaStreaming_RTSPStreaming)

Test Purpose: To verify that stream over RTSP protocol was successfully established between Client and Device using RTSP commands and then successfully stopped.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with RTSP SETUP request with transport parameter as "RTP/AVP/TCP" and which does not contain Require header with "onvif-replay" value and which is not tunneled in HTTP present.
- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetStreamUri** for Media Service with **trt:StreamSetup\trt:Transport\trt:Protocol** element value equals to "RTSP".

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetStreamUri** request message for media profile with Stream Type element with "RTP-Unicast" OR "RTP-Multicast" value and Transport Protocol element with "RTSP" value.

2. Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
3. Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
4. Device responds with code RTSP 200 OK.
5. Client invokes **RTSP SETUP** request with **Transport** tag in RTSP header that contains "RTP/AVP/TCP" and without "onvif-replay" Require header to set media session parameters.
6. Device responds with code RTSP 200 OK.
7. Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
8. Device responds with code RTSP 200 OK.
9. Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.
10. If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:

Note: If no **GetStreamUri** (Media Service) corresponding to detected RTSP session found, the test will be assumed as NOT DETECTED.

PASS -

- Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
 - [S1] It contains **Transport** request header field with value is equal to "RTP/AVP/TCP" (transport=RTP, profile=AVP, lower-transport=TCP) (see [RFC 2326]) AND
 - [S2] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
 - [S3] It is not tunneled in HTTP AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S4] It has RTSP 200 response code AND
- There is Client **RTSP DESCRIBE** request in Test Procedure fulfills the following requirements:
 - [S5] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S6] It invoked before the Client **RTSP SETUP** request AND

- [S7] It is not tunneled in HTTP AND
- Device response on the **RTSP DESCRIBE** request fulfills the following requirements:
 - [S8] SDP packet contains media type with Control URL that was used to send **RTSP SETUP** (see [RFC 2326, C.1.1 Control URL]) AND
 - [S9] It has RTSP 200 response code AND
- There is a Device **GetStreamUri** request in Test Procedure fulfills the following requirements:
 - [S10] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S11] It invoked before the Client **RTSP DESCRIBE** request AND
 - [S12] **trt:StreamSetup/tt:Transport/tt:Protocol** element value is equal to "RTSP"
- Device response on the **GetStreamUri** request fulfills the following requirements:
 - [S13] It has HTTP 200 response code AND
 - [S14] It contains **trt:MediaUri/tt:Uri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request AND
- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:
 - [S15] It is invoked for the same RTSP session as the Client **RTSP SETUP** request AND
 - [S16] It invoked after the Client **RTSP SETUP** request AND
 - [S17] RTSP address that was used to send it is correspond to any media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S18] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
 - [S19] It is not tunneled in HTTP AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:
 - [S20] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
 - [S21] It invoked for the same RTSP session as the Client **RTSP SETUP** request AND

- [S22] It invoked after the Client **RTSP PLAY** request AND
- [S23] RTSP address that was used to send it is correspond to any media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- [S24] It is not tunneled in HTTP AND
- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:
 - [S25] It has RTSP 200 response code.

FAIL -

- The Client failed PASS criteria.

5.4.6 STREAMING OVER UDP

Test Label: Media Streaming - UDP

Test Case ID: MEDIASTREAMING-4

Feature Under Test: Streaming Over UDP (MediaStreaming_UDP)

Test Purpose: To verify that stream over UDP protocol was successfully established between Client and Device using RTSP commands and then successfully stopped.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with RTSP SETUP request with transport parameter as "RTP/AVP/UDP" or "RTP/AVP" and which does not contain Require header with "onvif-replay" value present.
- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetStreamUri** for Media Service with **trt:StreamSetup/tt:Transport/tt:Protocol** element value equals to "UDP".

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetStreamUri** request message for media profile with Stream Type element with "RTP-Unicast" OR "RTP-Multicast" value and Transport Protocol element with "UDP" value.
2. Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.

3. Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
4. Device responds with code RTSP 200 OK.
5. Client invokes **RTSP SETUP** request with **Transport** tag in RTSP header that contains "RTP/AVP/UDP" or "RTP/AVP" and without "onvif-replay" Require header to set media session parameters.
6. Device responds with code RTSP 200 OK.
7. Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
8. Device responds with code RTSP 200 OK.
9. Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.
10. If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:

Note: If no **GetStreamUri** (Media Service) corresponding to detected RTSP session found, the test will be assumed as NOT DETECTED.

PASS -

- Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
 - [S1] It contains **Transport** request header field with value is equal to "RTP/AVP/UDP" OR "RTP/AVP" (transport=RTP, profile=AVP, lower-transport=TCP or skipped) (see [RFC 2326]) AND
 - [S2] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S3] It has RTSP 200 response code AND
- There is Client **RTSP DESCRIBE** request in Test Procedure fulfills the following requirements:
 - [S4] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S5] It invoked before the Client **RTSP SETUP** request AND
- Device response on the **RTSP DESCRIBE** request fulfills the following requirements:
 - [S6] SDP packet contains media type with Control URL that was used to send **RTSP SETUP** (see [RFC 2326, C.1.1 Control URL]) AND

- [S7] It has RTSP 200 response code AND
- There is a Device **GetStreamUri** request in Test Procedure fulfills the following requirements:
 - [S8] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S9] It invoked before the Client **RTSP DESCRIBE** request AND
 - [S10] **trt:StreamSetup/tt:Transport/tt:Protocol** element value is equal to "UDP"
- Device response on the **GetStreamUri** request fulfills the following requirements:
 - [S11] It contains **trt:MediaUri/tt:Uri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request AND
 - [S12] It has HTTP 200 response code AND
- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:
 - [S13] It invoked for the same RTSP session as the Client **RTSP SETUP** request AND
 - [S14] It invoked after the Client **RTSP SETUP** request AND
 - [S15] RTSP address that was used to send it is correspond to any media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S16] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:
 - [S17] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
 - [S18] It invoked for the same RTSP session as the Client **RTSP SETUP** request AND
 - [S19] It invoked after the Client **RTSP PLAY** request AND
 - [S20] RTSP address that was used to send it is correspond to any media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:

- [S21] It has RTSP 200 response code.

FAIL -

- The Client failed PASS criteria.

5.4.7 STREAMING OVER HTTP

Test Label: Media Streaming - HTTP

Test Case ID: MEDIASTREAMING-5

Feature Under Test: Streaming Over HTTP (MediaStreaming_HTTP)

Test Purpose: To verify that stream over HTTP protocol was successfully established between Client and Device using RTSP commands and then successfully stopped.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with RTSP SETUP request with transport parameter as "RTP/AVP/TCP" and which does not contain Require header with "onvif-replay" value and which is tunneled in HTTP present.
- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetStreamUri** for Media Service with **trt:StreamSetup/tt:Transport/tt:Protocol** element value equals to "HTTP".

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetStreamUri** request message for media profile with Stream Type element with "RTP-Unicast" OR "RTP-Multicast" value and Transport Protocol element with "RTSP" value.
2. Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
3. Client invokes **RTSP DESCRIBE** request in HTTP tunnel to retrieve media stream description.
4. Device responds with code RTSP 200 OK.
5. Client invokes **RTSP SETUP** request without "onvif-replay" Require header in HTTP tunnel with **Transport** tag in RTSP header that contains "RTP/AVP/TCP" to set media session parameters.

6. Device responds with code RTSP 200 OK.
7. Client invokes **RTSP PLAY** request without "onvif-replay" Require header in HTTP tunnel to start media stream.
8. Device responds with code RTSP 200 OK.
9. Client invokes **RTSP TEARDOWN** request in HTTP tunnel to terminate the RTSP session.
10. If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:

Note: If no **GetStreamUri** (Media Service) corresponding to detected RTSP session found, the test will be assumed as NOT DETECTED.

PASS -

- Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
 - [S1] It contains **Transport** request header field with value is equal to "RTP/AVP/TCP" (transport=RTP, profile=AVP, lower-transport=TCP) (see [RFC 2326]) AND
 - [S2] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
 - [S3] It is tunneled in HTTP AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S4] It has RTSP 200 response code AND
- There is Client **RTSP DESCRIBE** request in Test Procedure fulfills the following requirements:
 - [S5] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S6] It invoked before the Client **RTSP SETUP** request AND
 - [S7] It is tunneled in HTTP AND
- Device response on the **RTSP DESCRIBE** request fulfills the following requirements:
 - [S8] SDP packet contains media type with Control URL that was used to send **RTSP SETUP** (see [RFC 2326, C.1.1 Control URL]) AND
 - [S9] It has RTSP 200 response code AND
- There is a Device **GetStreamUri** request in Test Procedure fulfills the following requirements:

- [S10] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
- [S11] It invoked before the Client **RTSP DESCRIBE** request AND
- [S12] **trt:StreamSetup/tt:Transport/tt:Protocol** element value is equal to "HTTP"
- Device response on the **GetStreamUri** request fulfills the following requirements:
 - [S13] It has HTTP 200 response code AND
 - [S14] It contains **trt:MediaUri/tt:Uri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request AND
- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:
 - [S15] It invoked for the same RTSP session as the Client **RTSP SETUP** request AND
 - [S16] It invoked after the Client **RTSP SETUP** request AND
 - [S17] RTSP address that was used to send it is correspond to any media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S18] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
 - [S19] It is tunneled in HTTP AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:
 - [S20] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
 - [S21] It invoked for the same RTSP session as the Client **RTSP SETUP** request AND
 - [S22] It invoked after the Client **RTSP PLAY** request AND
 - [S23] RTSP address that was used to send it is correspond to any media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S24] It is tunneled in HTTP AND
- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:

- [S25] It has RTSP 200 response code.

FAIL -

- The Client failed PASS criteria.

5.5 Video Streaming Test Cases

5.5.1 Feature Level Requirement:

Validated Feature: Video Streaming (VideoStreaming)

Check Condition based on Device Features: Real Time Streaming (Media Service) is supported by Device.

Required Number of Devices: 3

Profile S Requirement: Mandatory

5.5.2 Expected Scenarios Under Test:

1. Client connects to Device to initiate Video Streaming of a specific encoding type.
2. Client is considered as supporting Video Streaming if the following conditions are met:
 - Client is able to initiate and retrieve a video stream with MJPEG encoding type (when the device doesn't support optional encoding features) OR
 - Client is able to initiate and retrieve a video stream with MJPEG encoding AND support all optional encodings (when the device supports optional encodings).
3. Client is considered as NOT supporting Video Streaming if ANY of the following is TRUE:
 - MJPEG Video Streaming attempts detected have failed OR
 - (when the device supports optional MPEG4 or H264 encodings) EITHER MPEG4 Video Streaming attempts detected have failed OR H264 Video Streaming attempts detected have failed.

5.5.3 MJPEG VIDEO STREAMING

Test Label: Video Streaming - MJPEG

Test Case ID: VIDEOSTREAMING-1

Feature Under Test: MJPEG Video Streaming (VideoStreaming_MJPEGStreaming)

Test Purpose: To verify that the Client is able to initiate and retrieve a video stream with MJPEG encoding type.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with Video Streaming of MJPEG encoding type.
- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetStreamUri** for Media Service.
- Device supports JPEG encoding for Video Streaming.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetStreamUri** request message for media profile that contains Video Source Configuration and Video Encoder Configuration with JPEG Encoding value. GetStreamUri request is set for RTP-Unicast/UDP OR RTP-Multicast/UDP OR RTP/RTSP/TCP OR RTP-Unicast/RTSP/HTTP/TCP transport.
2. Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
3. Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
4. Device responds with code RTSP 200 OK and SDP information with Media Type: "video" and with encoding name "JPEG" or with payload type number "26".
5. Client invokes **RTSP SETUP** request without "onvif-replay" Require header and with transport parameter element to set media session parameters for JPEG video streaming.
6. Device responds with code RTSP 200 OK.
7. Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
8. Device responds with code RTSP 200 OK.
9. Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.
10. If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:

Note: If no **GetStreamUri** (Media Service) corresponding to detected RTSP session found, the test will be assumed as NOT DETECTED.

Note: RTSP requests and RTSP response could be tunneled in HTTP if RTP-Unicast/RTSP/HTTP/TCP transport is used.

PASS -

- There is Client **RTSP DESCRIBE** request in Test Procedure
- Device response on the **RTSP DESCRIBE** request fulfills the following requirements:
 - [S1] It has RTSP 200 response code AND
 - [S2] IF SDP packet contains media type "video" (m=video) with sessions attribute "rtptime" THEN encoding name is "JPEG"
 - [S3] ELSE IF SDP packet contains media type "video" (m=video) without sessions attribute "rtptime" THEN payload type number is "26" (see [RFC 2435]) AND
- There is Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
 - [S4] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S5] It invoked after the Client **RTSP DESCRIBE** request AND
 - [S6] RTSP address that was used to send **RTSP SETUP** is correspond to corresponding media Control URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S7] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S8] It has RTSP 200 response code AND
- There is a Device response on the **GetStreamUri** request in Test Procedure fulfills the following requirements:
 - [S9] It has HTTP 200 response code AND
 - [S10] It received for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S11] It received before the Client **RTSP DESCRIBE** request AND
 - [S12] It contains **trt:MediaUri\rt:Uri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request AND
- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:
 - [S13] It invoked for the same RTSP session as the Client **RTSP SETUP** request AND

- [S14] It invoked after the Client **RTSP SETUP** request AND
- [S15] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- [S16] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:
 - [S17] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
 - [S18] It invoked for the same RTSP session as the Client **RTSP SETUP** request AND
 - [S19] It invoked after the Client **RTSP PLAY** request AND
 - [S20] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:
 - [S21] It has RTSP 200 response code.

FAIL -

- The Client failed PASS criteria.

5.5.4 MPEG4 VIDEO STREAMING

Test Label: Video Streaming - MPEG4

Test Case ID: VIDEOSTREAMING-2

Feature Under Test: MPEG4 Video Streaming (VideoStreaming_MPEG4Streaming)

Test Purpose: To verify that the Client is able to initiate and retrieve a video stream with MPEG4 encoding type.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with Video Streaming of MPEG4 encoding type.
- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetStreamUri** for Media Service.
- Device supports MPEG4 encoding for Video Streaming.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetStreamUri** request message for media profile that contains Video Source Configuration and Video Encoder Configuration with MPEG4 Encoding value. GetStreamUri request is set for RTP-Unicast/UDP OR RTP-Multicast/UDP OR RTP/RTSP/TCP OR RTP-Unicast/RTSP/HTTP/TCP transport.
2. Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
3. Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
4. Device responds with code RTSP 200 OK and SDP information with Media Type: "video" and with encoding name "MP4V-ES".
5. Client invokes **RTSP SETUP** request without "onvif-replay" Require header and with transport parameter element to to set media session parameters for MPEG4 video streaming.
6. Device responds with code RTSP 200 OK.
7. Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
8. Device responds with code RTSP 200 OK.
9. Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.
10. If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:

Note: If no **GetStreamUri** (Media Service) corresponding to detected RTSP session found, the test will be assumed as NOT DETECTED.

Note: RTSP requests and RTSP response could be tunneled in HTTP if RTP-Unicast/RTSP/HTTP/TCP transport is used.

PASS -

- There is Client **RTSP DESCRIBE** request in Test Procedure
- Device response on the **RTSP DESCRIBE** request fulfills the following requirements:
 - [S1] It has RTSP 200 response code AND
 - [S2] SDP packet contains media type "video" (m=video) with sessions attribute "rtpmap" with encoding name "MP4V-ES" (see [RFC 3016], item 5.2 SDP usage of MPEG-4 Visual) AND
- There is Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
 - [S3] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S4] It invoked after the Client **RTSP DESCRIBE** request AND
 - [S5] RTSP address that was used to send **RTSP SETUP** is correspond to corresponding media Control URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S6] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S7] It has RTSP 200 response code AND
- There is a Device response on the **GetStreamUri** request in Test Procedure fulfills the following requirements:
 - [S8] It has HTTP 200 response code AND
 - [S9] It received for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S10] It received before the Client **RTSP DESCRIBE** request AND
 - [S11] It contains **trt:MediaUri|tt:Uri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request AND
- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:
 - [S12] It invoked for the same RTSP session as the Client **RTSP SETUP** request AND
 - [S13] It invoked after the Client **RTSP SETUP** request AND
 - [S14] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND

- [S15] It does not contain **Require** request header field with value is equal to "onvif-replay"
AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:
 - [S16] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
 - [S17] It invoked the same RTSP session as the Client **RTSP SETUP** request AND
 - [S18] It invoked after the Client **RTSP PLAY** request AND
 - [S19] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:
 - [S20] It has RTSP 200 response code.

FAIL -

- The Client failed PASS criteria.

5.5.5 H264 VIDEO STREAMING

Test Label: Video Streaming - H264

Test Case ID: VIDEOSTREAMING-3

Feature Under Test: H264 Video Streaming (VideoStreaming_H264Streaming)

Test Purpose: To verify that the Client is able to initiate and retrieve a video stream with H264 encoding type.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with Video Streaming of H264 encoding type.
- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetStreamUri** for Media Service.
- Device supports H264 encoding for Video Streaming.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetStreamUri** request message for media profile that contains Video Source Configuration and Video Encoder Configuration with H264 Encoding value. GetStreamUri request is set for RTP-Unicast/UDP OR RTP-Multicast/UDP OR RTP/RTSP/TCP OR RTP-Unicast/RTSP/HTTP/TCP transport.
2. Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
3. Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
4. Device responds with code RTSP 200 OK and SDP information with Media Type: "video" and with encoding name "H264".
5. Client invokes **RTSP SETUP** request without "onvif-replay" Require header and with transport parameter element to set media session parameters for H264 video streaming.
6. Device responds with code RTSP 200 OK.
7. Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
8. Device responds with code RTSP 200 OK.
9. Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.
10. If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:

Note: If no **GetStreamUri** (Media Service) corresponding to detected RTSP session found, the test will be assumed as NOT DETECTED.

Note: RTSP requests and RTSP response could be tunneled in HTTP if RTP-Unicast/RTSP/HTTP/TCP transport is used.

PASS -

- There is Client **RTSP DESCRIBE** request in Test Procedure
- Device response on the **RTSP DESCRIBE** request fulfills the following requirements:
 - [S1] It has RTSP 200 response code AND
 - [S2] SDP packet contains media type "video" (m=video) with sessions attribute "rtmap" with encoding name "H264" (see [RFC 3984], item 8.2.1. Mapping of MIME Parameters to SDP) AND
- There is Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:

- [S3] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
- [S4] It invoked after the Client **RTSP DESCRIBE** request AND
- [S5] RTSP address that was used to send **RTSP SETUP** is correspond to corresponding media Control URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- [S6] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S7] It has RTSP 200 response code AND
- There is a Device response on the **GetStreamUri** request in Test Procedure fulfills the following requirements:
 - [S8] It has HTTP 200 response code AND
 - [S9] It received for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S10] It received before the Client **RTSP DESCRIBE** request AND
 - [S11] It contains **trt:MediaUri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request AND
- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:
 - [S12] It invoked for the same RTSP session as the Client **RTSP SETUP** request AND
 - [S13] It invoked after the Client **RTSP SETUP** request AND
 - [S14] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S15] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:
 - [S16] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
 - [S17] It invoked for the same RTSP session as the Client **RTSP SETUP** request AND

- [S18] It invoked after the Client **RTSP PLAY** request AND
- [S19] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:
 - [S20] It has RTSP 200 response code.

FAIL -

- The Client failed PASS criteria.

5.6 Video Encoder Configurations Test Cases

5.6.1 Feature Level Requirement:

Validated Feature: Video Encoder Configurations (VideoEncoderConfigurations)

Check Condition based on Device Features: Media Service is supported by Device.

Required Number of Devices: 3

Profile S Requirement: Mandatory

5.6.2 Expected Scenarios Under Test:

1. Client connects to Device to modify Video Encoder Configurations.
2. Client is considered as supporting Video Encoder Configurations if the following conditions are met:
 - Device returns a valid response to **GetVideoEncoderConfigurations** operations AND
 - Device returns a valid response to **GetVideoEncoderConfiguration** operations AND
 - Client is able to retrieve video encoder configuration options using **GetVideoEncoderConfigurationOptions** operation AND
 - Client is able to change video encoder configuration settings using **SetVideoEncoderConfiguration** operation.

3. Client is considered as NOT supporting Video Encoder Configurations if ANY of the following is TRUE:
 - No Valid Device Response to **GetVideoEncoderConfigurations** request if detected OR
 - No Valid Device Response to **GetVideoEncoderConfiguration** request if detected OR
 - No valid responses for **GetVideoEncoderConfigurationOptions** request OR
 - No valid responses for **SetVideoEncoderConfiguration** request.

5.6.3 LIST VIDEO ENCODER CONFIGURATIONS

Test Label: Video Encoder Configurations - list all existing video encoder configurations

Test Case ID: VIDEOENCODERCONFIGURATIONS-1

Feature Under Test: List Video Encoder Configurations
(VideoEncoderConfigurations_GetVideoEncoderConfigurations)

Test Purpose: To verify that list of all existing video encoder configurations from Device is received by Client using the GetVideoEncoderConfigurations operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetVideoEncoderConfigurations operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetVideoEncoderConfigurations request message to retrieve complete list of available video encoder configurations from Device.
2. Device responds with code HTTP 200 OK and GetVideoEncoderConfigurationsResponse message.

Test Result:

PASS -

- Client **GetVideoEncoderConfigurations** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetVideoEncoderConfigurations** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:GetVideoEncoderConfigurations** AND

- Device response on the **GetVideoEncoderConfigurations** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **trt:GetVideoEncoderConfigurationsResponse**.

FAIL -

- The Client failed PASS criteria.

5.6.4 GET SPECIFIC VIDEO ENCODER CONFIGURATION

Test Label: Video Encoder Configurations - gets a specific encoder configuration

Test Case ID: VIDEOENCODERCONFIGURATIONS-2

Feature Under Test: Get Specific Video Encoder Configuration
(VideoEncoderConfigurations_GetVideoEncoderConfiguration)

Test Purpose: To verify that Client is able to retrieve a specific encoder configuration from Device by using the GetVideoEncoderConfiguration operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetVideoEncoderConfiguration operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetVideoEncoderConfiguration request message with specified ConfigurationToken.
2. Device responds with code HTTP 200 OK and GetVideoEncoderConfigurationResponse.

Test Result:**PASS -**

- Client **GetVideoEncoderConfiguration** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetVideoEncoderConfiguration** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetVideoEncoderConfiguration>" tag after the "<Body>" tag AND

- [S2] "<GetVideoEncoderConfiguration>" includes tag: "<ConfigurationToken>" with non-empty string value of "Token=*" parameter AND
- [S3] Device response contains "HTTP/* 200 OK" AND
- [S4] Device response contains "<GetVideoEncoderConfigurationResponse>" tag.

FAIL -

- The Client failed PASS criteria.

5.6.5 GET VIDEO ENCODER CONFIGURATION OPTIONS

Test Label: Video Encoder Configuration - Get Video Encoder Configuration Options

Test Case ID: VIDEOENCODERCONFIGURATIONS-3

Feature Under Test: Get Video Encoder Configuration Options
(VideoEncoderConfigurations_GetVideoEncoderConfigurationOptions)

Test Purpose: To verify that Client is able to get video encoder configuration options provided by Device using the **GetVideoEncoderConfigurationOptions** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetVideoEncoderConfigurationOptions** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetVideoEncoderConfigurationOptions** request message to retrieve video encoder configuration options for the Device.
2. Device responds with code HTTP 200 OK and **GetVideoEncoderConfigurationOptionsResponse** message.

Test Result:**PASS -**

- Client **GetVideoEncoderConfigurationOptions** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetVideoEncoderConfigurationOptions** request in Test Procedure fulfills the following requirements:

- [S1] **soapenv:Body** element has child element **trt:GetVideoEncoderConfigurationOptions** AND
- [S2] If it contains **trt:ConfigurationToken** element THEN it has non-empty string value AND
- [S3] If it contains **trt:ProfileToken** element THEN it has non-empty string value AND
- Device response to the **GetVideoEncoderConfigurationOptions** request fulfills the following requirements:
 - [S4] It has HTTP 200 response code AND
 - [S5] **soapenv:Body** element has child element **trt:GetVideoEncoderConfigurationOptionsResponse**.

FAIL -

- The Client failed PASS criteria.

5.6.6 SET VIDEO ENCODER CONFIGURATION

Test Label: Configure Video Encoder Configuration - Set Video Encoder Configuration

Test Case ID: VIDEOENCODERCONFIGURATIONS-4

Feature Under Test: Set Video Encoder Configuration
(VideoEncoderConfigurations_SetVideoEncoderConfiguration)

Test Purpose: To verify that Client is able to change video encoder configuration provided by Device using the **SetVideoEncoderConfiguration** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetVideoEncoderConfiguration** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetVideoEncoderConfiguration** request message to change video encoder configuration on the Device.
2. Device responds with code HTTP 200 OK and **SetVideoEncoderConfigurationResponse** message.

Test Result:

PASS -

- Client **SetVideoEncoderConfiguration** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetVideoEncoderConfiguration** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:SetVideoEncoderConfiguration** AND
 - [S2] **trt:SetVideoEncoderConfiguration/trt:Configuration/@token** element has non-empty string value AND
- Device response to the **SetVideoEncoderConfiguration** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **trt:SetVideoEncoderConfigurationResponse**.

FAIL -

- The Client failed PASS criteria.

5.7 Multiple Video Sources Test Cases

5.7.1 Feature Level Requirement:

Validated Feature: Multiple Video Sources (MultipleVideoSources)

Check Condition based on Device Features: Real Time Streaming (Media Service) is supported by Device.

Required Number of Devices: 3

Profile S Requirement: Mandatory

5.7.2 Expected Scenarios Under Test:

1. Client connects to Device to get all Video Sources.
2. Client obtains video streaming for each Video Source provided by a Device.
3. Client is considered as supporting Multiple Video Sources if the following conditions are met:

- Client is able to get profile list by using **GetProfiles** operation AND
 - Client is able to to initiate and retrieve video stream for each Video Source provided by a Device using **GetStreamUri** command and **RTSP** commands.
4. Client is considered as NOT supporting Multiple Video Sources if ANY of the following is TRUE:
- No Valid Device Response to **GetProfiles** request OR
 - Client is unable to initiate and retrieve video streaming for at least one Video Source provided by a Device.

5.7.3 STREAMING WITH ALL VIDEO SOURCES DETECTED IN GET PROFILES

Test Label: Multiple Video Sources - Streaming with all Video Sources detected in GetProfilesResponse

Test Case ID: MULTIPLEVIDEOSOURCES-1

Feature Under Test: Streaming For Video Sources From GetProfiles (MultipleVideoSources_StreamingForVideoSourcesFromGetProfiles)

Test Purpose: To verify that Client is able to obtain video streaming for each video source provided by a Device in GetProfiles responses.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with video streaming present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetProfiles** request messages to retrieve complete list of available media profiles with video source configurations from Device.
2. Device responds with code HTTP 200 OK and **GetProfilesResponse** message.
3. Client initiate video streaming for each Video Source token detected in GetProfilesResponse:
 - Client selects existing media profile with required Video Source token or modifies media profile to have required Video Source token or creates media profile with required Video Source token.

- Client invokes **GetStreamUri** request for this media profile.
- Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
- Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
- Device responds with code RTSP 200 OK and SDP information with Media Type: "video".
- Client invokes **RTSP SETUP** request without "onvif-replay" Require header and with transport parameter element to set media session parameters for video streaming.
- Device responds with code RTSP 200 OK.
- Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
- Device responds with code RTSP 200 OK.
- Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.
- If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:**PASS -**

- For each Video Source Token listed by HelperGetVideoSourcesListFromGetProfiles (see [Annex A.1](#)) there is a video stream in Test Procedure that fulfills the following requirements:
 - There is a Client **GetStreamUri** request that fulfills the following requirements:
 - [S1] It invoked for the media profile which contains Video Source Configuration with this Video Source Token (see [Annex A.2](#) HelperGetVideoSourceTokenUsedForStreaming to get video source token from media profile) AND
 - Device response on the **GetStreamUri** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **trt:GetStreamUriResponse** AND
 - There is a RTSP session in Test Procedure that fulfills the following requirements:
 - [S5] It invoked for the uri from **GetStreamUri** response AND
 - [S6] It started video streaming according to HelperFindVideoStreamingForGetStreamUri (see [Annex A.3](#))

FAIL -

- The Client failed PASS criteria.

6 Test Cases for Profile Conditional Features

6.1 Event Handling Test Cases

6.1.1 Feature Level Requirement:

Validated Feature: Event Handling (EventHandling)

Check Condition based on Device Features: Pull Point Notification OR WS Basic Notification OR Profile S OR Metadata under Media2 service is supported by Device.

Required Number of Devices: 3

Profile S Requirement: Conditional

Profile G Requirement: Conditional

Profile A Requirement: Mandatory

Profile C Requirement: Mandatory

Profile T Requirement: Mandatory

Profile D Requirement: Mandatory

6.1.2 Expected Scenarios Under Test:

1. Client connects to Device to initiate Event Handling.
2. Client is considered as supporting Event Handling if the following conditions are met:
 - Client is able to handle the Pull Point Event mechanism OR
 - Client is able to handle the Base Notification Event mechanism OR
 - Client is able to handle the Metadata Streaming by supporting EventHandling_MetadataStreamingUsingMedia feature (please see [EVENTHANDLING-4 METADATA STREAMING USING MEDIA](#) section) OR Media2_MetadataStreaming_MetadataStreamingUsingMedia2 feature (please see [MEDIA2_METADATASTREAMING-1 METADATA STREAMING USING MEDIA2](#) section).
3. Client is considered as NOT supporting Event Handling if the following is TRUE:
 - All Pull Point attempts detected have failed AND

- All Base Notification attempts detected have failed AND
- All Metadata Streaming attempts detected have failed.

6.1.3 PULLPOINT

Test Label: Event Handling - Pull Point

Test Case ID: EVENTHANDLING-1

Feature Under Test: Pull Point (EventHandling_PullPoint)

Test Purpose: To verify that the Client is able to retrieve events using Pull Point.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with Pull Point event type.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes CreatePullPointSubscription message.
2. Device responds with code HTTP 200 OK and CreatePullPointSubscriptionResponse message.
3. Client invokes PullMessages command with Timeout and MessageLimit elements.
4. Device responds with code HTTP 200 OK and PullMessagesResponse message.

Test Result:

PASS -

- Client **CreatePullPointSubscription** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **CreatePullPointSubscription** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<CreatePullPointSubscription>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND
 - [S3] Device response contains "<CreatePullPointSubscriptionResponse>" tag AND

- Client **PullMessages** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **PullMessages** request in Test Procedure fulfills the following requirements:
 - [S4] Client request contains "<PullMessages>" tag after the "<Body>" tag AND
 - [S7] Device response contains "HTTP/* 200 OK" AND
 - [S8] Device response contains "<PullMessagesResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.1.4 BASE NOTIFICATION

Test Label: Event Handling - Basic Notification

Test Case ID: EVENTHANDLING-2

Feature Under Test: Base Notification (EventHandling_WSBaseNotification)

Test Purpose: To verify that the Client is able to retrieve events using WS-Base Notification.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with Basic Notification event type.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes Subscribe message with ConsumerReference element.
2. Device responds with code HTTP 200 OK and SubscribeResponse message.

Test Result:**PASS -**

- Client **Subscribe** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **Subscribe** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<Subscribe>" tag after the "<Body>" tag AND
 - [S4] Device response contains "HTTP/* 200 OK" AND

- [S5] Device response contains "<SubscribeResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.1.5 METADATA STREAMING USING MEDIA

Test Label: Event Handling - Metadata Streaming Using Media Streaming

Test Case ID: EVENTHANDLING-4

Feature Under Test: Metadata Streaming (EventHandling_MetadataStreamingUsingMedia)

Test Purpose: To verify that the Client is able to retrieve the Metadata Streaming using Media Service.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with Metadata Streaming event type using Media Service.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetStreamUri** request message for Media service for media profile that contains Video Source Configuration and Metadata Configuration. GetStreamUri request is set for RTP-Unicast/UDP OR RTP-Multicast/UDP OR RTP/RTSP/TCP OR RTP-Unicast/RTSP/HTTP/TCP transport.
2. Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
3. Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
4. Device responds with code RTSP 200 OK and SDP information with Media Type: "application" and with encoding name "vnd.onvif.metadata" or "vnd.onvif.metadata.gzip" or "vnd.onvif.metadata.exi.onvif" or "vnd.onvif.metadata.exi.ext".
5. Client invokes **RTSP SETUP** request without "onvif-replay" Require header and with transport parameter element to to set media session parameters for metadata streaming.
6. Device responds with code RTSP 200 OK.
7. Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
8. Device responds with code RTSP 200 OK.
9. Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.

10. If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:

Note: RTSP requests and RTSP response could be tunneled in HTTP if RTP-Unicast/RTSP/HTTP/TCP transport is used.

PASS -

- There is Client **RTSP DESCRIBE** request in Test Procedure
- Device response on the **RTSP DESCRIBE** request fulfills the following requirements:
 - [S1] It has RTSP 200 response code AND
 - [S2] SDP packet contains media type "application" (m=application) with sessions attribute "rtptime" with encoding name "vnd.onvif.metadata" OR "vnd.onvif.metadata.gzip" OR "vnd.onvif.metadata.exi.onvif" OR "vnd.onvif.metadata.exi.ext" (see ONVIF Streaming Spec) AND
- There is Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
 - [S3] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S4] It invoked after the Client **RTSP DESCRIBE** request AND
 - [S5] RTSP address that was used to send **RTSP SETUP** is correspond to corresponding media Control URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S6] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S7] It has RTSP 200 response code AND
- There is a Device response on the **GetStreamUri** request invoked for Media Service in Test Procedure fulfills the following requirements:
 - [S8] It has HTTP 200 response code AND
 - [S9] It received for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S10] It received before the Client **RTSP DESCRIBE** request AND
 - [S11] It contains **trt:MediaUri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request AND

- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:
 - [S12] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S13] It invoked after the Client **RTSP SETUP** request AND
 - [S14] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S15] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:
 - [S16] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
 - [S17] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S18] It invoked after the Client **RTSP PLAY** request AND
 - [S19] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:
 - [S20] It has RTSP 200 response code.

FAIL -

- The Client failed PASS criteria.

6.2 Keep Alive for Pull Point Event Handling Test Cases

6.2.1 Feature Level Requirement:

Validated Feature: Keep Alive for Pull Point Event Handling
(KeepAliveForPullPointEventHandling)

Check Condition based on Device Features: Pull Point Notification is supported by Device.

Required Number of Devices: 3

Profile A Requirement: Mandatory

Profile C Requirement: Mandatory

Profile S Requirement: Conditional

Profile G Requirement: Conditional

Profile T Requirement: Optional

6.2.2 Expected Scenarios Under Test:

1. Client connects to Device to initiate Pull Point Event Handling.
2. Client is considered as supporting Keep Alive for Pull Point Event Handling if the following conditions are met:
 - Client supports `EventHandling_Pullpoint` feature (please see [EVENTHANDLING-1 PULLPOINT](#) section) AND
 - Client is able to renew pull point subscription using **Renew** operation OR **PullMessages** operation mechanism.
3. Client is considered as NOT supporting Keep Alive for Pull Point Event Handling if the following is TRUE:
 - No valid responses for **Renew** request AND for **CreatePullPointSubscription** request in the case if **PullMessages** used for keep alive OR
 - No valid responses for **Renew** request if detected OR
 - No valid responses for **CreatePullPointSubscription** request in the case if **PullMessages** used for keep alive if detected OR
 - **Renew** request was invoked to address which was not specified in `tev:SubscriptionReference/wsa:Address` element of corresponding **CreatePullPointSubscriptionResponse** message.

6.2.3 PULLPOINT

Test Label: Event Handling - Pull Point

Test Case ID: EVENTHANDLING-1

Feature Under Test: Pull Point (EventHandling_PullPoint)

Test Purpose: To verify that the Client is able to retrieve events using Pull Point.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with Pull Point event type.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes CreatePullPointSubscription message.
2. Device responds with code HTTP 200 OK and CreatePullPointSubscriptionResponse message.
3. Client invokes PullMessages command with Timeout and MessageLimit elements.
4. Device responds with code HTTP 200 OK and PullMessagesResponse message.

Test Result:

PASS -

- Client **CreatePullPointSubscription** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **CreatePullPointSubscription** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<CreatePullPointSubscription>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND
 - [S3] Device response contains "<CreatePullPointSubscriptionResponse>" tag AND
- Client **PullMessages** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **PullMessages** request in Test Procedure fulfills the following requirements:
 - [S4] Client request contains "<PullMessages>" tag after the "<Body>" tag AND
 - [S7] Device response contains "HTTP/* 200 OK" AND
 - [S8] Device response contains "<PullMessagesResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.2.4 RENEW

Test Label: Advanced Pull Point Event Handling - Renew

Test Case ID: KEEPALIVEFORPULLPOINTEVENTHANDLING-1

Feature Under Test: Renew (KeepAliveForPullPointEventHandling_Renew)

Test Purpose: To verify that the Client is able to use **Renew** operation as keep alive for Pull Point subscription.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **Renew** operations present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **CreatePullPointSubscription** message.
2. Device responds with code HTTP 200 OK and **CreatePullPointSubscriptionResponse** message.
3. Client invokes **Renew** message to valid address recieved in **CreatePullPointSubscriptionResponse** message for the created Pull Point subscription with valid address recieved in **CreatePullPointSubscriptionResponse** message.
4. Device responds with code HTTP 200 OK and **RenewResponse** message.

Test Result:

PASS -

- Client **Renew** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **Renew** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **wsnt:Renew** AND
- Device response on the **Renew** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **wsnt:RenewResponse** AND

- There is a Device response on the **CreatePullPointSubscription** request in Test Procedure fulfills the following requirements:
 - [S4] It has HTTP 200 response code AND
 - [S5] It received for the same Device as for the Client **Renew** request AND
 - [S6] It received before the Client **Renew** request AND
 - [S7] It contains **tev:SubscriptionReference\wsa:Address** element which is equal to HTTP address that was used to send the **Renew** request.

FAIL -

- The Client failed PASS criteria.

6.2.5 PULL MESSAGES AS KEEP ALIVE

Test Label: Advanced Pull Point Event Handling - Pull Messages as Keep Alive

Test Case ID: KEEPALIVEFORPULLPOINTEVENTHANDLING-2

Feature Under Test: Pull Messages as Keep Alive
(KeepAliveForPullPointEventHandling_PullMessagesAsKeepAlive)

Test Purpose: To verify that the Client is able to use **PullMessages** operation as keep alive for Pull Point subscription.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **CreatePullPointSubscription** operations without **tev:InitialTerminationTime** element present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **CreatePullPointSubscription** message.
2. Device responds with code HTTP 200 OK and **CreatePullPointSubscriptionResponse** message without **tev:InitialTerminationTime** element.

Test Result:

PASS -

- Client **CreatePullPointSubscription** request messages are valid according to XML Schemas listed in [Namespaces](#) AND

- Client **CreatePullPointSubscription** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tev:CreatePullPointSubscription** AND
 - [S2] It does not contain **tev:InitialTerminationTime** element AND
- Device response on the **CreatePullPointSubscription** request fulfills the following requirements:
 - [S3] It has HTTP 200 response code AND
 - [S4] **soapenv:Body** element has child element **tev:CreatePullPointSubscriptionResponse**.

FAIL -

- The Client failed PASS criteria.

6.3 Discovery Test Cases

6.3.1 Feature Level Requirement:

Validated Feature: Discovery (Discovery)

Check Condition based on Device Features: Discovery

Required Number of Devices: 3

Profile S Requirement: Conditional

Profile C Requirement: Conditional

Profile G Requirement: Conditional

Profile A Requirement: Mandatory

Profile T Requirement: Mandatory

Profile D Requirement: Mandatory

Profile M Requirement: Mandatory

6.3.2 Expected Scenarios Under Test:

1. Client sends Probe message to multicast IP address 239.255.255.250 and port 3702 to locate services on a local network.

2. Client is considered as supporting Discovery if the following conditions are met:
 - Probe request detected AND at least one ProbeMatch response detected
3. Client is considered as NOT supporting Discovery if the following is TRUE:
 - No Valid Device Response to Probe request.

6.3.3 WS-DISCOVERY

Test Label: Discovery - WS-Discovery

Test Case ID: DISCOVERY-1

Feature Under Test: WS-Discovery (Discovery_WSDiscovery)

Test Purpose: To verify that Client is able to send Probe request and receive ProbeMatch response from Device.

Pre-Requisite:

- The Network Trace Capture files contain at least one Client Probe request to multicast IP address and one ProbeMatch response from Device directly to the Client.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes Probe request message to multicast IP address 239.255.255.250 and port 3702.
2. Device sends ProbeMatch message directly to the Client.

Test Result:

PASS -

- Client **Probe** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **Probe** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<Action>" tag after the "<Header>" tag AND
 - [S2] "<Action>" includes URL address which ends with "Probe" value AND
 - [S3] Client request contains "<MessageID>" with non-empty string value AND
 - [S4] Client request contains "<Probe>" tag after the "<Body>" tag AND

- [S5] Device response message contains "<ProbeMatches>" tag after the "<Body>" tag.

FAIL -

- The Client failed PASS criteria.

6.4 Network Video Transmitter Discovery Type Filter Test Cases

6.4.1 Feature Level Requirement:

Validated Feature: NVT Discovery Type Filter (NVTDiscoveryTypeFilter)

Check Condition based on Device Features: Network Video Transmitter Discovery Type is supported by Device.

Required Number of Devices: 3

Profile S Requirement: Conditional

Profile A Requirement: None

Profile C Requirement: None

Profile G Requirement: None

Profile T Requirement: None

6.4.2 Expected Scenarios Under Test:

1. Client sends Probe message to multicast IPv4 address 239.255.255.250 or multicast IPv6 address [FF02::C] and port 3702 with Types filter that contains **dn:NetworkVideoTransmitter** or with skipped Types filter.
2. Client is considered as supporting Network Video Transmitter Discovery Type if the following conditions are met:
 - **Probe** Client message that fulfills the following requirement is detected:
 - Types filter contains dn:NetworkVideoTransmitter or empty or skipped AND
 - Probe is sent to multicast IPv4 address 239.255.255.250 or multicast IPv6 address [FF02::C] AND

- Probe is sent to UDP port 3702 AND
 - There is **ProbeMatch** Device message that correspond to Client **Probe**.
3. Client is considered as NOT supporting Network Video Transmitter Discovery Type if the following is TRUE:
 - No valid Device **ProbeMatch** message that is correspond to Client **Probe** message.

6.4.3 NVT DISCOVERY TYPE FILTER

Test Label: Discovery - Network Video Transmitter Discovery Type Filter

Test Case ID: NVTDISCOVERYTYPEFILTER-1

Feature Under Test: Network Video Transmitter Discovery Type Filter
(NVTDiscoveryTypeFilter_NetworkVideoTransmitterFilter)

Test Purpose: To verify that Client is able to discover devices with Network Video Transmitter Discovery Type.

Pre-Requisite:

- The Network Trace Capture files contains at least one Client Probe message that does not filter out devices with Network Video Transmitter Discovery Type that is sent to multicast WS-Discovery address.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes Probe request message to multicast IPv4 address 239.255.255.250 or multicast IPv6 address [FF02::C] and port 3702 with **Types** that contains dn:NetworkVideoTransmitter.
2. Device sends ProbeMatch message to the Client.

Test Result:

PASS -

- Client **Probe** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **Probe** request in Test Procedure fulfills the following requirements:
 - [S1] It is sent to 239.255.255.250 IPv4 address OR [FF02::C] IPv6 address AND

- [S2] It is sent to 3702 UDP port AND
- [S3] **soapenv:Envelope/soapenv:Header** element has child element **wsadis:Action** AND
- [S4] **wsadis:Action** includes URL address which ends with "Probe" value AND
- [S5] **soapenv:Envelope/soapenv:Header** element has child element **wsadis:MessageID** with non-empty string value AND
- [S6] **soapenv:Body** element has child element **d:Probe** AND
- [S7] IF **d:Probe** element has child element **d:Types** THEN it contains value is equal to **dn:NetworkVideoTransmitter** OR empty string value AND
- [S8] There is Device **ProbeMatches** message in test procedure that fulfills the following requirements:
 - [S9] **soapenv:Body** element has child element **d:ProbeMatches** AND
 - [S10] **soapenv:Envelope/soapenv:Header/wsadis:RelatesTo** element value is equal to **soapenv:Envelope/soapenv:Header/wsadis:MessageID** value in **Probe** message AND

PASS WITH WARNING -

- **d:Probe/d:Types** element is skipped OR
- **d:Probe/d:Types** element has empty string value.

FAIL -

- The Client failed PASS criteria.

6.5 Network Configuration Test Cases

6.5.1 Feature Level Requirement:

Validated Feature: Network Configuration (NetworkConfiguration)

Check Condition based on Device Features: Network Configuration

Required Number of Devices: 3

Profile A Requirement: Conditional

Profile C Requirement: Conditional

Profile D Requirement: Mandatory

Profile G Requirement: Conditional

Profile S Requirement: Conditional

Profile T Requirement: Mandatory

Profile M Requirement: Mandatory

6.5.2 Expected Scenarios Under Test:

1. Client connects to Device to configure network settings.
2. Client is considered as supporting Network Configuration if the following conditions are met:
 - Client is able to list network interfaces of Device using the GetNetworkInterfaces operation AND
 - Client is able to set network interfaces of Device using the SetNetworkInterfaces operation AND
 - Client is able to list default gateway of Device using the GetNetworkDefaultGateway operation AND
 - Client is able set default gateway of Device using the SetNetworkDefaultGateway operation.
3. Client is considered as NOT supporting Network Configuration if ANY of the following is TRUE:
 - No Valid Device Response to GetNetworkInterfaces request OR
 - No Valid Device Response to SetNetworkInterfaces request OR
 - No Valid Device Response to GetNetworkDefaultGateway request OR
 - No Valid Device Response to SetNetworkDefaultGateway request.

6.5.3 GET NETWORK INTERFACES

Test Label: Network Configuration - Get Network Interfaces

Test Case ID: NETWORKCONFIGURATION-1

Feature Under Test: Get Network Interfaces (NetworkConfiguration_GetNetworkInterfaces)

Test Purpose: To verify that Client is able to list network interfaces of Device using the GetNetworkInterfaces operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetNetworkInterfaces operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetNetworkInterfaces request message to get network interface configuration from Device.
2. Device responds with code HTTP 200 OK and GetNetworkInterfacesResponse message.

Test Result:

PASS -

- Client **GetNetworkInterfaces** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetNetworkInterfaces** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetNetworkInterfaces>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND
 - [S3] Device response contains "<GetNetworkInterfacesResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.5.4 SET NETWORK INTERFACES

Test Label: Network Configuration - Set Network Interfaces

Test Case ID: NETWORKCONFIGURATION-2

Feature Under Test: Set Network Interfaces (NetworkConfiguration_SetNetworkInterfaces)

Test Purpose: To verify that Client is able to set network interfaces of Device using the SetNetworkInterfaces operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetNetworkInterfaces operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetNetworkInterfaces request message to set the network interface configuration on Device.
2. Device responds with code HTTP 200 OK and SetNetworkInterfacesResponse message.

Test Result:**PASS -**

- Client **SetNetworkInterfaces** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetNetworkInterfaces** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<SetNetworkInterfaces>" tag after the "<Body>" tag AND
 - [S2] "<SetNetworkInterfaces>" includes tag: "<InterfaceToken>" with non-empty string value of specific token AND
 - [S4] Device response contains "HTTP/* 200 OK" AND
 - [S5] Device response contains "<SetNetworkInterfacesResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.5.5 GET NETWORK DEFAULT GATEWAY

Test Label: Network Configuration - Get Network Default Gateway

Test Case ID: NETWORKCONFIGURATION-3

Feature Under Test: Get Network Default Gateway
(NetworkConfiguration_GetNetworkDefaultGateway)

Test Purpose: To verify that Client is able to list default gateway of Device using the GetNetworkDefaultGateway operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetNetworkDefaultGateway operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetNetworkDefaultGateway request message to get the default gateway settings from Device.
2. Device responds with code HTTP 200 OK and GetNetworkDefaultGatewayResponse message.

Test Result:

PASS -

- Client **GetNetworkDefaultGateway** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetNetworkDefaultGateway** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetNetworkDefaultGateway>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND
 - [S3] Device response contains "<GetNetworkDefaultGatewayResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.5.6 SET NETWORK DEFAULT GATEWAY

Test Label: Network Configuration - Set Network Default Gateway

Test Case ID: NETWORKCONFIGURATION-4

Feature Under Test: Set Network Default Gateway
(NetworkConfiguration_SetNetworkDefaultGateway)

Test Purpose: To verify that Client is able to set default gateway of Device using the SetNetworkDefaultGateway operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetNetworkDefaultGateway operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetNetworkDefaultGateway request message to set the default gateway settings on Device.
2. Device responds with code HTTP 200 OK and SetNetworkDefaultGatewayResponse message.

Test Result:**PASS -**

- Client **SetNetworkDefaultGateway** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetNetworkDefaultGateway** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<SetNetworkDefaultGateway>" tag after the "<Body>" tag AND
 - [S2] "<SetNetworkDefaultGateway>" includes tag: EITHER "<IPv4Address>" OR "<IPv6Address>" with specific IP address value AND
 - [S3] Device response contains "HTTP/* 200 OK" AND
 - [S4] Device response contains "<SetNetworkDefaultGatewayResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.6 System Test Cases

6.6.1 Feature Level Requirement:

Validated Feature: System (System)

Check Condition based on Device Features: None

Required Number of Devices: 3

Profile A Requirement: Conditional

Profile C Requirement: Conditional

Profile G Requirement: Conditional

Profile S Requirement: Conditional

Profile T Requirement: Conditional

Profile D Requirement: Conditional

Profile M Requirement: Conditional

6.6.2 Expected Scenarios Under Test:

1. Client connects to Device to get information, such as manufacturer, model, firmware version and etc.
2. Client is considered as supporting System if the following conditions are met:
 - Client is able to list Device information using the GetDeviceInformation operation.
3. Client is considered as NOT supporting System if ANY of the following is TRUE:
 - No Valid Device Response to GetDeviceInformation request.

6.6.3 GET DEVICE INFORMATION

Test Label: System - Get Device Information

Test Case ID: SYSTEM-1

Feature Under Test: Get Device Information (System_GetDeviceInformation)

Test Purpose: To verify that Client is able to list Device information using the GetDeviceInformation operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetDeviceInformation operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetDeviceInformation request message to list Device information.
2. Device responds with code HTTP 200 OK and GetDeviceInformationResponse message.

Test Result:**PASS -**

- Client **GetDeviceInformation** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetDeviceInformation** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetDeviceInformation>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND
 - [S3] Device response contains "<GetDeviceInformationResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.7 User Handling Test Cases

6.7.1 Feature Level Requirement:

Validated Feature: User Handling (UserHandling)

Check Condition based on Device Features: User Configuration

Required Number of Devices: 3

Profile A Requirement: Mandatory

Profile S Requirement: Conditional

Profile C Requirement: Conditional

Profile G Requirement: Conditional

Profile T Requirement: Conditional

Profile D Requirement: Conditional

6.7.2 Expected Scenarios Under Test:

1. Client connects to Device to create, list, modify and delete users.
2. Client is considered as supporting User Handling if the following conditions are met:

- Client is able to create users on Device using the CreateUsers operation AND
 - Client is able to list existing users of Device using the GetUsers operation AND
 - Client is able to modify users on Device using the SetUser operation AND
 - Client is able to delete users from Device using the DeleteUsers operation.
3. Client is considered as NOT supporting System if ANY of the following is TRUE:
- No Valid Device Response to CreateUsers request (except SOAP fault: **soapenv:Receiver/ter:Action/ter:TooManyUsers**) OR
 - No Valid Device Response to GetUsers request OR
 - No Valid Device Response to SetUser request (except SOAP fault: **soapenv:Sender/ter:InvalidArgVal/ter:FixedUser**) OR
 - No Valid Device Response to DeleteUsers request (except SOAP fault: **soapenv:Sender/ter:InvalidArgVal/ter:FixedUser**).

6.7.3 CREATE USERS

Test Label: User Handling - CreateUsers

Test Case ID: USERHANDLING-1

Feature Under Test: Create Users (UserHandling_CreateUsers)

Test Purpose: To verify that Client is able to create users on Device using the CreateUsers operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with CreateUsers operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes CreateUsers request message to create new users and corresponding credentials on Device.
2. Device responds with code HTTP 200 OK and CreateUsersResponse message.

Test Result:

PASS -

- Client **CreateUsers** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **CreateUsers** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<CreateUsers>" tag after the "<Body>" tag AND
 - [S2] "<CreateUsers>" includes tag: "<User>" AND
 - [S3] "<User>" includes tag: "<Username>" with non-empty string value AND
 - [S4] "<User>" includes tag: "<Password>" with non-empty string value AND
 - [S5] If Device response contains "HTTP/* 200 OK" THEN it contains "<CreateUsersResponse>" tag, ELSE it contains **soapenv:Fault** with **soapenv:Receiver/ter:Action/ter:TooManyUsers** fault code.

FAIL -

- The Client failed PASS criteria.

6.7.4 GET USERS

Test Label: User Handling - GetUsers

Test Case ID: USERHANDLING-2

Feature Under Test: Get Users (UserHandling_GetUsers)

Test Purpose: To verify that Client is able to list existing users of Device using the GetUsers operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetUsers operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetUsers request message to list registered users and their user levels.
2. Device responds with code HTTP 200 OK and GetUsersResponse message.

Test Result:

PASS -

- Client **GetUsers** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetUsers** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetUsers>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND
 - [S3] Device response contains "<GetUsersResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.7.5 SET USER

Test Label: User Handling - SetUser

Test Case ID: USERHANDLING-3

Feature Under Test: Set User (UserHandling_SetUser)

Test Purpose: To verify that Client is able to modify users on Device using the SetUser operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetUser operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetUser request message to update the authentication settings on Device.
2. Device responds with code HTTP 200 OK and SetUserResponse message.

Test Result:**PASS -**

- Client **SetUser** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetUser** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<SetUser>" tag after the "<Body>" tag AND

- [S2] "<SetUser>" includes tag: "<User>" AND
- [S3] "<User>" includes tag: "<Username>" with non-empty string value AND
- [S4] If Device response contains "HTTP/* 200 OK" THEN it contains "<SetUserResponse>" tag, ELSE it contains **soapenv:Fault** with **soapenv:Sender/ter:InvalidArgVal/ter:FixedUser** fault code.

FAIL -

- The Client failed PASS criteria.

6.7.6 DELETE USERS

Test Label: User Handling - DeleteUsers

Test Case ID: USERHANDLING-4

Feature Under Test: Delete Users (UserHandling_DeleteUsers)

Test Purpose: To verify that Client is able to delete users from Device using the DeleteUsers operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with DeleteUsers operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes DeleteUsers request message to delete specific users from Device.
2. Device responds with code HTTP 200 OK and DeleteUsersResponse message.

Test Result:**PASS -**

- Client **DeleteUsers** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **DeleteUsers** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<DeleteUsers>" tag after the "<Body>" tag AND
 - [S2] "<DeleteUsers>" includes tag: "<Username>" with non-empty string value AND

- [S3] If Device response contains "HTTP/* 200 OK" THEN it contains "<DeleteUsersResponse>" tag, ELSE it contains **soapenv:Fault** with **soapenv:Sender/ter:InvalidArgVal/ter:FixedUser** fault code.

FAIL -

- The Client failed PASS criteria.

6.8 Relay Outputs Test Cases

6.8.1 Feature Level Requirement:

Validated Feature: Relay Outputs (RelayOutputs)

Check Condition based on Device Features: Relay Outputs (Device Management Service) is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

6.8.2 Expected Scenarios Under Test:

1. Client connects to Device to list, configure and trigger relay outputs using Device Management service.
2. Client is considered as supporting Relay Outputs if the following conditions are met:
 - Client is able to list available relay outputs using the GetRelayOutputs operation using Device Management service AND
 - Client is able to trigger relay output using the SetRelayOutputState operation using Device Management service AND
 - Client is able to set settings of relay output in EITHER "Bistable" OR "Monostable" mode using the SetRelayOutputSettings operation using Device Management service.
3. Client is considered as NOT supporting Relay Outputs if ANY of the following is TRUE:
 - No Valid Device Response to GetRelayOutputs request to Device Management service OR
 - No Valid Device Response to SetRelayOutputState request to Device Management service OR

- No Valid Device Response to SetRelayOutputSettings requests to Device Management service for BOTH "Bistable" AND "Monostable" mode.

6.8.3 GET RELAY OUTPUTS

Test Label: Relay Output - Get Relay Outputs

Test Case ID: RELAYOUTPUTS-1

Feature Under Test: Get Relay Outputs (RelayOutputs_GetRelayOutputs)

Test Purpose: To verify that Client is able to list available relay outputs using the GetRelayOutputs operation for Device Management Service.

Test Purpose: To verify that relay outputs provided by Device is received by Client using the **GetRelayOutputs** operation using Device Management Service.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetRelayOutputs** operation for Device Management Service present.
- Client supports Capabilities feature.
- The Client Test Tool retrieves Device Management Service address from device's response on GetServices or GetCapabilities Client request.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetRelayOutputs** request message to Device Management Service to retrieve relay outputs from the Device.
2. Device responds with code HTTP 200 OK and **GetRelayOutputsResponse** message.

Test Result:

PASS -

- Client **GetRelayOutputs** request messages to Device Management Service are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetRelayOutputs** request to Device Management Service in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tds:GetRelayOutputs** AND
- Device response on the **GetRelayOutputs** request fulfills the following requirements:

- [S2] It has HTTP 200 response code AND
- [S3] **soapenv:Body** element has child element **tds:GetRelayOutputsResponse**.

FAIL -

- The Client failed PASS criteria.

6.8.4 SET RELAY OUTPUT STATE

Test Label: Relay Output - Set Relay Output State

Test Case ID: RELAYOUTPUTS-2

Feature Under Test: Set Relay Output State (RelayOutputs_SetRelayOutputState)

Test Purpose: To verify that Client is able to trigger a relay output using the **SetRelayOutputState** operation for Device Management Service.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetRelayOutputState** operation for Device Management Service present.
- Client supports Capabilities feature.
- The Client Test Tool retrieves Device Management Service address from device's response on GetServices or GetCapabilities Client request.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetRelayOutputState** request message to Device Management Service to trigger a relay output on the Device.
2. Device responds with code HTTP 200 OK and **SetRelayOutputStateResponse** message.

Test Result:**PASS -**

- Client **SetRelayOutputState** request messages to Device Management Service are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetRelayOutputState** request to Device Management Service in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tds:SetRelayOutputState** AND

- [S2] **tds:SetRelayOutputState\tds:RelayOutputToken** element has non-empty string value AND
- Device response on the **SetRelayOutputState** request fulfills the following requirements:
 - [S3] It has HTTP 200 response code AND
 - [S4] **soapenv:Body** element has child element **tds:SetRelayOutputStateResponse**.

FAIL -

- The Client failed PASS criteria.

6.8.5 SET RELAY OUTPUT SETTINGS BISTABLE MODE

Test Label: Relay Outputs - SetRelayOutputSettings Bistable Mode

Test Case ID: RELAYOUTPUTS-3

Feature Under Test: Set Relay Output Settings Bistable Mode
(RelayOutputs_SetRelayOutputBistable)

Test Purpose: To verify that Client is able to set settings of relay output in "Bistable" mode using the **SetRelayOutputSettings** operation for Device Management Service.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetRelayOutputSettings** operation for Device Management Service with **tds:SetRelayOutputSettings\tds:Properties\lt:Mode** element value is equal to "Bistable" present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetRelayOutputSettings** request message to Device Management Service to set setting of relay output in "Bistable" mode.
2. Device responds with code HTTP 200 OK and **SetRelayOutputSettingsResponse** message.

Test Result:

NOTE: If Client SetRelayOutputSettings request message does not contain "Bistable" value of Mode element then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **SetRelayOutputSettings** request messages to Device Management Service are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetRelayOutputSettings** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tds:SetRelayOutputSettings** AND
 - [S2] **tds:SetRelayOutputSettings\tds:RelayOutputToken** element has non-empty string value AND
 - [S2] **tds:SetRelayOutputSettings\tds:Properties\tt:Mode** element value is equal to "Bistable" AND
- Device response on the **SetRelayOutputSettings** request fulfills the following requirements:
 - [S4] It has HTTP 200 response code AND
 - [S5] **soapenv:Body** element has child element **tds:SetRelayOutputSettingsResponse**.

FAIL -

- The Client failed PASS criteria.

6.8.6 SET RELAY OUTPUT SETTINGS MONOSTABLE MODE

Test Label: Relay Outputs - SetRelayOutputSettings Monostable Mode

Test Case ID: RELAYOUTPUTS-4

Feature Under Test: Set Relay Output Settings Monostable Mode
(RelayOutputs_SetRelayOutputMonostable)

Test Purpose: To verify that Client is able to set settings of relay output in "Monostable" mode using the **SetRelayOutputSettings** operation for Device Management Service.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetRelayOutputSettings** operation for Device Management Service with **tds:SetRelayOutputSettings\tds:Properties\tt:Mode** element value is equal to "Monostable" present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetRelayOutputSettings** request message to Device Management Service to set setting of relay output in "Monostable" mode.

2. Device responds with code HTTP 200 OK and **SetRelayOutputSettingsResponse** message.

Test Result:

NOTE: If Client SetRelayOutputSettings request message does not contain "Monostable" value of Mode element then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **SetRelayOutputSettings** request messages to Device Management Service are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetRelayOutputSettings** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tds:SetRelayOutputSettings** AND
 - [S2] **tds:SetRelayOutputSettings\tds:RelayOutputToken** element has non-empty string value AND
 - [S2] **tds:SetRelayOutputSettings\tds:Properties\tds:Mode** element value is equal to "Monostable" AND
- Device response on the **SetRelayOutputSettings** request fulfills the following requirements:
 - [S4] It has HTTP 200 response code AND
 - [S5] **soapenv:Body** element has child element **tds:SetRelayOutputSettingsResponse**.

FAIL -

- The Client failed PASS criteria.

6.9 NTP Test Cases

6.9.1 Feature Level Requirement:

Validated Feature: NTP (NTP)

Check Condition based on Device Features: NTP is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

Profile T Requirement: Conditional

6.9.2 Expected Scenarios Under Test:

1. Client connects to Device to configure synchronization of time using NTP servers on Device.
2. Client is considered as supporting NTP if the following conditions are met:
 - Client is able to get the NTP settings from Device using the GetNTP operation AND
 - Client is able to set the NTP settings on Device using the SetNTP operation.
3. Client is considered as NOT supporting NTP if ANY of the following is TRUE:
 - No Valid Device Response to GetNTP request OR
 - No Valid Device Response to SetNTP request.

6.9.3 GET NTP

Test Label: NTP - GetNTP

Test Case ID: NTP-1

Feature Under Test: Get NTP (NTP_GetNTP)

Test Purpose: To verify that Client is able to get the NTP settings from Device using the GetNTP operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetNTP operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetNTP request message to get current settings of NTP servers on Device.
2. Device responds with code HTTP 200 OK and GetNTPResponse message.

Test Result:

PASS -

- Client **GetNTP** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetNTP** request in Test Procedure fulfills the following requirements:

- [S1] Client request contains "<GetNTP>" tag after the "<Body>" tag AND
- [S2] Device response contains "HTTP/* 200 OK" AND
- [S3] Device response contains "<GetNTPResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.9.4 SET NTP

Test Label: NTP - SetNTP

Test Case ID: NTP-2

Feature Under Test: Set NTP (NTP_SetNTP)

Test Purpose: To verify that Client is able to set the NTP settings on Device using the SetNTP operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetNTP operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetNTP request message to set the NTP servers settings on Device.
2. Device responds with code HTTP 200 OK and SetNTPResponse message.

Test Result:**PASS -**

- Client **SetNTP** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetNTP** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<SetNTP>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND
 - [S3] Device response contains "<SetNTPResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.10 Dynamic DNS Test Cases

6.10.1 Feature Level Requirement:

Validated Feature: Dynamic DNS (DynamicDns)

Check Condition based on Device Features: Dynamic DNS is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

6.10.2 Expected Scenarios Under Test:

1. Client connects to Device to configure Dynamic DNS settings.
2. Client is considered as supporting Dynamic DNS if the following conditions are met:
 - Client is able to get the Dynamic DNS settings from Device using the GetDynamicDNS operation AND
 - Client is able to set the Dynamic DNS settings on Device using the SetDynamicDNS operation.
3. Client is considered as NOT supporting Dynamic DNS if ANY of the following is TRUE:
 - No Valid Device Response to GetDynamicDNS request OR
 - No Valid Device Response to SetDynamicDNS request.

6.10.3 GET DYNAMIC DNS SETTINGS

Test Label: Dynamic DNS - GetDynamicDNS

Test Case ID: DYNAMICDNS-1

Feature Under Test: Get Dynamic DNS (DynamicDns_GetDynamicDnsSettings)

Test Purpose: To verify that Client is able get the dynamic DNS settings from Device using the GetDynamicDNS operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetDynamicDNS operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetDynamicDNS request message to get the dynamic DNS settings from Device.
2. Device responds with code HTTP 200 OK and GetDynamicDNSResponse message.

Test Result:**PASS -**

- Client **GetDynamicDNS** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetDynamicDNS** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetDynamicDNS>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND
 - [S3] Device response contains "<GetDynamicDNSResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.10.4 SET DYNAMIC DNS SETTINGS

Test Label: Dynamic DNS - SetDynamicDNS

Test Case ID: DYNAMICDNS-2

Feature Under Test: Set Dynamic DNS (DynamicDns_SetDynamicDnsSettings)

Test Purpose: To verify that Client is able set the dynamic DNS settings on Device using the SetDynamicDNS operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetDynamicDNS operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetDynamicDNS request message to set the dynamic DNS settings on Device.
2. Device responds with code HTTP 200 OK and SetDynamicDNSResponse message.

Test Result:**PASS -**

- Client **SetDynamicDNS** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetDynamicDNS** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<SetDynamicDNS>" tag after the "<Body>" tag AND
 - [S3] Device response contains "HTTP/* 200 OK" AND
 - [S4] Device response contains "<SetDynamicDNSResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.11 Zero Configuration Test Cases

6.11.1 Feature Level Requirement:

Validated Feature: Zero Configuration (ZeroConfiguration)

Check Condition based on Device Features: Zero Configuration is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

6.11.2 Expected Scenarios Under Test:

1. Client connects to Device to configure Zero Configuration settings.
2. Client is considered as supporting Zero Configuration if the following conditions are met:
 - Client is able to get the Zero Configuration settings from Device using the GetZeroConfiguration operation AND

- Client is able to set the Zero Configuration settings on Device using the SetZeroConfiguration operation.
3. Client is considered as NOT supporting Zero Configuration if ANY of the following is TRUE:
- No Valid Device Response to GetZeroConfiguration request OR
 - No Valid Device Response to SetZeroConfiguration request.

6.11.3 GET ZERO CONFIGURATION

Test Label: Zero Configuration - GetZeroConfiguration

Test Case ID: ZEROCONFIGURATION-1

Feature Under Test: Get Zero Configuration (ZeroConfiguration_GetZeroConfiguration)

Test Purpose: To verify that Client is able to get the Zero Configuration settings from Device using the GetZeroConfiguration operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetZeroConfiguration operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetZeroConfiguration request message to get the Zero Configuration settings from Device.
2. Device responds with code HTTP 200 OK and GetZeroConfigurationResponse message.

Test Result:

PASS -

- Client **GetZeroConfiguration** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetZeroConfiguration** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetZeroConfiguration>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND
 - [S3] Device response contains "<GetZeroConfigurationResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.11.4 SET ZERO CONFIGURATION

Test Label: Zero Configuration - SetZeroConfiguration

Test Case ID: ZEROCONFIGURATION-2

Feature Under Test: Set Zero Configuration (ZeroConfiguration_SetZeroConfiguration)

Test Purpose: To verify that Client is able to set the Zero Configuration settings on Device using the SetZeroConfiguration operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetZeroConfiguration operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetZeroConfiguration request message to set the Zero Configuration settings on Device.
2. Device responds with code HTTP 200 OK and SetZeroConfigurationResponse message.

Test Result:

PASS -

- Client **SetZeroConfiguration** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetZeroConfiguration** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<SetZeroConfiguration>" tag after the "<Body>" tag AND
 - [S2] "<SetZeroConfiguration>" includes tag: "<InterfaceToken>" with non-empty string value of specific token AND
 - [S3] Device response contains "HTTP/* 200 OK" AND
 - [S4] Device response contains "<SetZeroConfigurationResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.12 IP Address Filtering Test Cases

6.12.1 Feature Level Requirement:

Validated Feature: IP Address Filtering (IPAddressFiltering)

Check Condition based on Device Features: IP Filter is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

Profile C Requirement: Conditional

Profile A Requirement: Conditional

6.12.2 Expected Scenarios Under Test:

1. Client connects to Device to manage IP address filters.
2. Client is considered as supporting IP Address Filtering if the following conditions are met:
 - Client is able to get the IP address filter settings from Device using the GetIPAddressFilter operation AND
 - Client is able to set the IP address filter settings on Device using the SetIPAddressFilter operation AND
 - Client is able to add the IP address filter settings to Device using the AddIPAddressFilter operation AND
 - Client is able to delete the IP address filter settings from Device using the RemoveIPAddressFilter operation.
 - **NOTE:** Requests SetIPAddressFilter, AddIPAddressFilter and RemoveIPAddressFilter are permitted to use the IPv4 OR IPv6 protocol settings.
3. Client is considered as NOT supporting IP Address Filtering if ANY of the following is TRUE:
 - No Valid Device Response to GetIPAddressFilter request OR
 - No Valid Device Response to SetIPAddressFilter request OR
 - No Valid Device Response to AddIPAddressFilter request OR

- No Valid Device Response to RemoveIPAddressFilter request.
- NOTE: Requests SetIPAddressFilter, AddIPAddressFilter and RemoveIPAddressFilter should be deemed as failed if both IPv4 AND IPv6 protocol settings have No Valid Device Responses.

6.12.3 GET IP ADDRESS FILTER

Test Label: IP Address Filtering - GetIPAddressFilter

Test Case ID: IPADDRESSFILTERING-1

Feature Under Test: Get Ip Address Filter (IPAddressFiltering_GetIpAddressFilter)

Test Purpose: To verify that Client is able to get the IP address filter settings from Device using the GetIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetIPAddressFilter request message to get the IP address filter settings from Device.
2. Device responds with code HTTP 200 OK and GetIPAddressFilterResponse message.

Test Result:

PASS -

- Client **GetIPAddressFilter** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetIPAddressFilter** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetIPAddressFilter>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND
 - [S3] Device response contains "<GetIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.12.4 SET IPv4 ADDRESS FILTER

Test Label: IP Address Filtering - SetIPv4AddressFilter

Test Case ID: IPADDRESSFILTERING-2

Feature Under Test: Set IPv4 Address Filter (IPAddressFiltering_SetIPv4AddressFilter)

Test Purpose: To verify that Client is able to set the IP address filter settings on Device using the SetIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetIPAddressFilter request message to set the IP address filter settings on Device.
2. Device responds with code HTTP 200 OK and SetIPAddressFilterResponse message.

Test Result:

NOTE: If Client SetIPAddressFilter request message does not contain "<IPv4Address>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **SetIPAddressFilter** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetIPAddressFilter** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<SetIPAddressFilter>" tag after the "<Body>" tag AND
 - [S3] "<SetIPAddressFilter>" includes tag: "<IPv4Address>" AND
 - [S4] "<IPv4Address>" includes tag: "<Address>" with specific IPv4 address value AND
 - [S5] "<IPv4Address>" includes tag: "<PrefixLength>" with value range from "0" to "32" AND
 - [S6] Device response contains "HTTP/* 200 OK" AND
 - [S7] Device response contains "<SetIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.12.5 SET IPv6 ADDRESS FILTER

Test Label: IP Address Filtering - SetIPv6AddressFilter

Test Case ID: IPADDRESSFILTERING-3

Feature Under Test: Set IPv6 Address Filter (IPAddressFiltering_SetIPv6AddressFilter)

Test Purpose: To verify that Client is able to set the IP address filter settings on Device using the SetIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetIPAddressFilter request message to set the IP address filter settings on Device.
2. Device responds with code HTTP 200 OK and SetIPAddressFilterResponse message.

Test Result:

NOTE: If Client SetIPAddressFilter request message does not contain "<IPv6Address>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **SetIPAddressFilter** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetIPAddressFilter** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<SetIPAddressFilter>" tag after the "<Body>" tag AND
 - [S3] "<SetIPAddressFilter>" includes tag: "<IPv6Address>" AND
 - [S4] "<IPv6Address>" includes tag: "<Address>" with specific IPv6 address value AND
 - [S5] "<IPv6Address>" includes tag: "<PrefixLength>" with value range from "0" to "128" AND

- [S6] Device response contains "HTTP/* 200 OK" AND
- [S7] Device response contains "<SetIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.12.6 ADD IPv4 ADDRESS FILTER

Test Label: IP Address Filtering - AddIPv4AddressFilter

Test Case ID: IPADDRESSFILTERING-4

Feature Under Test: Add IPv4 Address Filter (IPAddressFiltering_AddIPv4AddressFilter)

Test Purpose: To verify that Client is able to add the IP address filter to Device using the AddIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with AddIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes AddIPAddressFilter request message to add the IP address filter on Device.
2. Device responds with code HTTP 200 OK and AddIPAddressFilterResponse message.

Test Result:

NOTE: If Client AddIPAddressFilter request message does not contain "<IPv4Address>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **AddIPAddressFilter** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **AddIPAddressFilter** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<AddIPAddressFilter>" tag after the "<Body>" tag AND
 - [S3] "<AddIPAddressFilter>" includes tag: "<IPv4Address>" AND
 - [S4] "<IPv4Address>" includes tag: "<Address>" with specific IPv4 address value AND

- [S5] "<IPv4Address>" includes tag: "<PrefixLength>" with value range from "0" to "32" AND
- [S6] Device response contains "HTTP/* 200 OK" AND
- [S7] Device response contains "<AddIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.12.7 ADD IPv6 ADDRESS FILTER

Test Label: IP Address Filtering - AddIPv6AddressFilter

Test Case ID: IPADDRESSFILTERING-5

Feature Under Test: Add IPv6 Address Filter (IPAddressFiltering_AddIPv6AddressFilter)

Test Purpose: To verify that Client is able to add the IP address filter to Device using the AddIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with AddIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes AddIPAddressFilter request message to add the IP address filter on Device.
2. Device responds with code HTTP 200 OK and AddIPAddressFilterResponse message.

Test Result:

NOTE: If Client AddIPAddressFilter request message does not contain "<IPv6Address>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **AddIPAddressFilter** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **AddIPAddressFilter** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<AddIPAddressFilter>" tag after the "<Body>" tag AND
 - [S3] "<AddIPAddressFilter>" includes tag: "<IPv6Address>" AND

- [S4] "<IPv6Address>" includes tag: "<Address>" with specific IPv6 address value AND
- [S5] "<IPv6Address>" includes tag: "<PrefixLength>" with value range from "0" to "128" AND
- [S6] Device response contains "HTTP/* 200 OK" AND
- [S7] Device response contains "<AddIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.12.8 REMOVE IPv4 ADDRESS FILTER

Test Label: IP Address Filtering - RemoveIPv4AddressFilter

Test Case ID: IPADDRESSFILTERING-6

Feature Under Test: Remove IPv4 Address Filter (IPAddressFiltering_RemoveIPv4AddressFilter)

Test Purpose: To verify that Client is able to delete the IP address filter from Device using the RemoveIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with RemoveIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes RemoveIPAddressFilter request message to delete the IP address filter from Device.
2. Device responds with code HTTP 200 OK and RemoveIPAddressFilterResponse message.

Test Result:

NOTE: If Client RemoveIPAddressFilter request message does not contain "<IPv4Address>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **RemoveIPAddressFilter** request messages are valid according to XML Schemas listed in [Namespaces](#) AND

- Client **RemoveIPAddressFilter** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<RemoveIPAddressFilter>" tag after the "<Body>" tag AND
 - [S3] "<RemoveIPAddressFilter>" includes tag: "<IPv4Address>" AND
 - [S4] "<IPv4Address>" includes tag: "<Address>" with specific IPv4 address value AND
 - [S5] "<IPv4Address>" includes tag: "<PrefixLength>" with value range from "0" to "32" AND
 - [S6] Device response contains "HTTP/* 200 OK" AND
 - [S7] Device response contains "<RemoveIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.12.9 REMOVE IPv6 ADDRESS FILTER

Test Label: IP Address Filtering - RemoveIPv6AddressFilter

Test Case ID: IPADDRESSFILTERING-7

Feature Under Test: Remove IPv6 Address Filter (IPAddressFiltering_RemoveIPv6AddressFilter)

Test Purpose: To verify that Client is able to delete the IP address filter from Device using the RemoveIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with RemoveIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes RemoveIPAddressFilter request message to delete the IP address filter from Device.
2. Device responds with code HTTP 200 OK and RemoveIPAddressFilterResponse message.

Test Result:

NOTE: If Client RemoveIPAddressFilter request message does not contain "<IPv6Address>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **RemoveIPAddressFilter** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **RemoveIPAddressFilter** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<RemoveIPAddressFilter>" tag after the "<Body>" tag AND
 - [S3] "<RemoveIPAddressFilter>" includes tag: "<IPv6Address>" AND
 - [S4] "<IPv6Address>" includes tag: "<Address>" with specific IPv6 address value AND
 - [S5] "<IPv6Address>" includes tag: "<PrefixLength>" with value range from "0" to "128" AND
 - [S6] Device response contains "HTTP/* 200 OK" AND
 - [S7] Device response contains "<RemoveIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.13 Multicast Streaming Test Cases

6.13.1 Feature Level Requirement:

Validated Feature: Multicast Streaming (MulticastStreaming)

Check Condition based on Device Features: RTP-Multicast/UDP (Media Service) is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

6.13.2 Expected Scenarios Under Test:

1. Client connects to Device and initiates Multicast Streaming using RTSP or using StartMultiCastStreaming and StopMultiCastStreaming operations.
2. Client is considered as supporting Multicast Streaming if the following conditions are met:
 - Able to start and stop a multicast stream by using Start/StopMulticastStreaming OR
 - Able to start and stop a multicast stream by using RTSP

3. Client is considered as NOT supporting Multicast Streaming if ANY of the following is TRUE:
 - If using Start/StopMulticastStreaming -> session never passed the PLAY state or was never terminated AND
 - If using RTSP -> RTSP session never passed the PLAY state or was never terminated

6.13.3 MULTICAST STREAMING USING RTSP

Test Label: Multicast Streaming - RTSP multicast setup

Test Case ID: MULTICASTSTREAMING-1

Feature Under Test: Multicast Streaming Using RTSP (MulticastStreaming_RTSPMulticast)

Test Purpose: To verify that the Client is able to setup and initiate a multicast stream with RTSP commands for stream control.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with RTSP SETUP request with transport parameter as "RTP/AVP/UDP;multicast" or "RTP/AVP;multicast" and without "onvif-replay" Require header present.
- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetStreamUri** for Media Service with Stream Type element with "RTP-Multicast" value and Transport Protocol element with "UDP" value.
- Device supports RTPMulticastUDP feature.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetStreamUri** request message for media profile with Stream Type element with "RTP-Multicast" value and Transport Protocol element with "UDP" value.
2. Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
3. Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
4. Device responds with code RTSP 200 OK.
5. Client invokes **RTSP SETUP** request without "onvif-replay" Require header and with **Transport** tag in RTSP header that contains "RTP/AVP/UDP;multicast" or "RTP/AVP;multicast" to set media session parameters.

6. Device responds with code RTSP 200 OK.
7. Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
8. Device responds with code RTSP 200 OK.
9. Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.
10. If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:**PASS -**

- Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
 - [S1] It contains **Transport** request header field with value is equal to "RTP/AVP/UDP" OR "RTP/AVP" and with "multicast" parameter value (transport=RTP, profile=AVP, lower-transport=TCP or skipped, parameter=multicast) (see [RFC 2326]) AND
 - [S2] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S3] It has RTSP 200 response code AND
- There is Client **RTSP DESCRIBE** request in Test Procedure that fulfills the following requirements:
 - [S4] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S5] It invoked before the Client **RTSP SETUP** request AND
 - [S6] SDP packet contains media type with Control URL that was used to send **RTSP SETUP** (see [RFC 2326, C.1.1 Control URL]) AND
- Device response on the **RTSP DESCRIBE** request fulfills the following requirements:
 - [S7] It has RTSP 200 response code AND
- There is a Device **GetStreamUri** request in Test Procedure that fulfills the following requirements:
 - [S8] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S9] It invoked before the Client **RTSP DESCRIBE** request AND

- [S10] **trt:StreamSetup/tt:Stream** element value is equal to "RTP-Multicast"
- [S11] **trt:StreamSetup/tt:Transport/tt:Protocol** element value is equal to "UDP"
- Device response on the **GetStreamUri** request fulfills the following requirements:
 - [S12] It has HTTP 200 response code AND
 - [S13] It contains **trt:MediaUri/tt:Uri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request AND
- There is Client **RTSP PLAY** request in Test Procedure that fulfills the following requirements:
 - [S14] It invoked for the same RTSP session as the Client **RTSP SETUP** request AND
 - [S15] It invoked after the Client **RTSP SETUP** request AND
 - [S16] RTSP address that was used to send it is correspond to any media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S17] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:
 - [S18] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure that fulfills the following requirements:
 - [S19] It invoked for the same RTSP session as the Client **RTSP SETUP** request AND
 - [S20] It invoked after the Client **RTSP PLAY** request AND
 - [S21] RTSP address that was used to send it is correspond to any media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:
 - [S22] It has RTSP 200 response code.

FAIL -

- The Client failed PASS criteria.

6.13.4 MULTICAST STREAMING USING SOAP

Test Label: Multicast Streaming - SOAP multicast setup

Test Case ID: MULTICASTSTREAMING-2

Feature Under Test: Multicast Streaming Using SOAP (MulticastStreaming_SOAPMulticast)

Test Purpose: To verify that the Client is able to setup and initiate a multicast stream with Start/ StopMulticastStreaming SOAP operations for stream control.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with StartMulticastStreaming operation for stream control.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes StartMulticastStreaming request message with non-empty ProfileToken element.
2. Device responds with code HTTP 200 OK and StartMulticastStreamingResponse message.
3. Client invokes StopMulticastStreaming request message with non-empty ProfileToken element.
4. Device responds with code HTTP 200 OK and StopMulticastStreamingResponse message.

Test Result:

NOTE: In case when StopMulticastStreaming command is detected and StartMulticastStreaming command is not detected then the test shall be deemed as "NOT DETECTED".

PASS -

- Client **StartMulticastStreaming** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **StartMulticastStreaming** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:StartMulticastStreaming** AND
 - [S2] **trt:StartMulticastStreaming/trt:ProfileToken** element has non-empty string value of specified profile token AND
- Device response on the **StartMulticastStreaming** request fulfills the following requirements:

- [S3] It has HTTP 200 response code AND
- [S4] **soapenv:Body** element has child element **trt:StartMulticastStreamingResponse**.
- Client **StopMulticastStreaming** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **StopMulticastStreaming** request in Test Procedure fulfills the following requirements:
 - [S5] **soapenv:Body** element has child element **trt:StopMulticastStreaming** AND
 - [S6] **trt:StopMulticastStreaming/trt:ProfileToken** element has non-empty string value of specified profile token AND
- Device response on the **StopMulticastStreaming** request fulfills the following requirements:
 - [S7] It has HTTP 200 response code AND
 - [S8] **soapenv:Body** element has child element **trt:StopMulticastStreamingResponse**.

FAIL -

- The Client failed PASS criteria.

6.14 Media Profile Configurations Test Cases

6.14.1 Feature Level Requirement:

Validated Feature: Media Profile Configurations (MediaProfileConfigurations)

Check Condition based on Device Features: Media Service is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

6.14.2 Expected Scenarios Under Test:

1. Client connects to Device to retrieve and/or create Media Profile Configuration.
2. Client is considered as supporting Media Profile Configuration if the following conditions are met:
 - Client shall be able to list available profiles using GetProfiles operation AND

- Client may be able to get profile using GetProfile operation AND
 - Client shall be able to create a media profile using the CreateProfile operation.
3. Client is considered as NOT supporting Media Profile Configuration if ANY of the following is TRUE:
- No Valid Device Response to GetProfiles request OR
 - No Valid Device Response to GetProfile request if detected OR
 - No Valid Device Response to CreateProfile request (except soap fault: maximumnumberofprofiles).

6.14.3 LIST AVAILABLE MEDIA PROFILES

Test Label: Media Profile Configurations - list available profiles

Test Case ID: MEDIAPROFILECONFIGURATIONS-1

Feature Under Test: List Available Media Profiles (MediaProfileConfigurations_ListMediaProfiles)

Test Purpose: To verify that list of media profiles from Device is received by Client using the GetProfiles operation.

Pre-Requirement:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetProfiles operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetProfiles request message to retrieve complete profiles list from Device.
2. Device responds with code HTTP 200 OK and GetProfilesResponse message.

Test Result:

PASS -

- Client **GetProfiles** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetProfiles** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:GetProfiles** AND

- Device response on the **GetProfiles** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **trt:GetProfilesResponse**.

FAIL -

- The Client failed PASS criteria.

6.14.4 GET SPECIFIC MEDIA PROFILE

Test Label: Media Profile Configurations - gets a specific media profile.

Test Case ID: MEDIAPROFILECONFIGURATIONS-2

Feature Under Test: Get Specific Media Profile (MediaProfileConfigurations_GetMediaProfile)

Test Purpose: To verify that Client is able to retrieve a specific media profile from Device by using the GetProfile operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetProfile operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetProfile request message to retrieve a specific media profile from Device.
2. Device responds with code HTTP 200 OK and GetProfileResponse message.

Test Result:**PASS -**

- Client **GetProfile** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetProfile** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetProfile>" tag after the "<Body>" tag AND
 - [S2] "<GetProfile>" includes tag: "<ProfileToken>" with non-empty string value of specific profile token AND
 - [S3] Device response contains "HTTP/* 200 OK" AND

- [S4] Device response contains "<GetProfileResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.14.5 CREATE A MEDIA PROFILE

Test Label: Media Profile Configurations - create a media profile

Test Case ID: MEDIAPROFILECONFIGURATIONS-3

Feature Under Test: Create a Media Profile (MediaProfileConfigurations_CreateMediaProfile)

Test Purpose: To verify that Client is able to create a new media profile on Device by using the CreateProfile operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with CreateProfile operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes CreateProfile request message to create a new empty profile structure with no configuration entities.
2. Device responds with code HTTP 200 OK and CreateProfileResponse message.

Test Result:**PASS -**

- Client **CreateProfile** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **CreateProfile** request in Test Procedure fulfills the following requirements:
 - Client **CreateProfile** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
 - Client **CreateProfile** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:CreateProfile** AND
 - [S2] **trt:CreateProfile/trt:Name** has non-empty string value AND

- [S3] If **trt:CreateProfile** contains **trt:Token** element, THEN it has non-empty string value AND
- Device response on the **CreateProfile** request fulfills the following requirements:
 - [S4] It has HTTP 200 response code AND
 - [S5] **soapenv:Body** element has child element **trt:CreateProfileResponse**.

FAIL -

- The Client failed PASS criteria.

6.15 Video Source Configurations Test Cases

6.15.1 Feature Level Requirement:

Validated Feature: Video Source Configurations (VideoSourceConfigurations)

Check Condition based on Device Features: Media Service is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

6.15.2 Expected Scenarios Under Test:

1. Client connects to Device to list, modify and add Video Source Configurations.
2. Client is considered as supporting Video Source Configurations if the following conditions are met:
 - Device returns a valid response to **GetVideoSourceConfigurations** operations AND
 - Device returns a valid response to **GetVideoSourceConfiguration** operations AND
 - Client is able to retrieve video source configuration options using **GetVideoSourceConfigurationOptions** operation AND
 - Client is able to change video source configuration settings using **SetVideoSourceConfiguration** operation AND
 - Client is able to retrieve list of video source configurations that are compatible with a certain media profile using **GetCompatibleVideoSourceConfigurations** operation AND

- Client is able to add video source configurations using **AddVideoSourceConfiguration** operation.
3. Client is considered as NOT supporting Video Source Configurations if ANY of the following is TRUE:
- No Valid Device Response to **GetVideoSourceConfigurations** request if detected OR
 - No Valid Device Response to **GetVideoSourceConfiguration** request if detected OR
 - No valid responses for **GetVideoSourceConfigurationOptions** request OR
 - No valid responses for **SetVideoSourceConfiguration** request OR
 - No valid responses for **GetCompatibleVideoSourceConfigurations** request OR
 - No valid responses for **AddVideoSourceConfiguration** request.

6.15.3 LIST VIDEO SOURCE CONFIGURATIONS

Test Label: Video Source Configurations - list available video source configurations

Test Case ID: VIDEOSOURCECONFIGURATIONS-1

Feature Under Test: List Video Source Configurations
(VideoSourceConfigurations_GetVideoSourceConfigurations)

Test Purpose: To verify that list of all existing video source configurations from Device is received by Client using the GetVideoSourceConfigurations operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetVideoSourceConfigurations operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetVideoSourceConfigurations request message to retrieve complete list of available video encoder configurations from Device.
2. Device responds with code HTTP 200 OK and GetVideoSourceConfigurationsResponse message.

Test Result:

PASS -

- Client **GetVideoSourceConfigurations** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetVideoSourceConfigurations** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:GetVideoSourceConfigurations** AND
- Device response on the **GetVideoSourceConfigurations** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **trt:GetVideoSourceConfigurationsResponse**.

FAIL -

- The Client failed PASS criteria.

6.15.4 GET SPECIFIC VIDEO SOURCE CONFIGURATION

Test Label: Video Source Configurations - gets a specific video source configuration

Test Case ID: VIDEOSOURCECONFIGURATIONS-2

Feature Under Test: Get Specific Video Source Configuration
(VideoSourceConfigurations_GetVideoSourceConfiguration)

Test Purpose: To verify that Client is able to retrieve a specific video source configuration from Device by using the GetVideoSourceConfiguration operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetVideoSourceConfiguration operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetVideoSourceConfiguration request message with specified ConfigurationToken.
2. Device responds with code HTTP 200 OK and GetVideoSourceConfigurationResponse message.

Test Result:

PASS -

- Client **GetVideoSourceConfiguration** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetVideoSourceConfiguration** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetVideoSourceConfiguration>" tag after the "<Body>" tag AND
 - [S2] "<GetVideoSourceConfiguration>" includes tag: "<ConfigurationToken>" with non-empty string value AND
 - [S3] Device response contains "HTTP/* 200 OK" AND
 - [S4] Device response contains "<GetVideoSourceConfigurationResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.15.5 GET VIDEO SOURCE CONFIGURATION OPTIONS

Test Label: Video Source Configuration - Get Video Source Configuration Options

Test Case ID: VIDEOSOURCECONFIGURATIONS-3

Feature Under Test: Get Video Source Configuration Options
(VideoSourceConfigurations_GetVideoSourceConfigurationOptions)

Test Purpose: To verify that Client is able to get video source configuration options provided by Device using the **GetVideoSourceConfigurationOptions** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetVideoSourceConfigurationOptions** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetVideoSourceConfigurationOptions** request message to retrieve video source configuration options for the Device.
2. Device responds with code HTTP 200 OK and **GetVideoSourceConfigurationOptionsResponse** message.

Test Result:

PASS -

- Client **GetVideoSourceConfigurationOptions** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetVideoSourceConfigurationOptions** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:GetVideoSourceConfigurationOptions** AND
 - [S2] If it contains **trt:ConfigurationToken** element THEN it has non-empty string value AND
 - [S3] If it contains **trt:ProfileToken** element THEN it has non-empty string value AND
- Device response to the **GetVideoSourceConfigurationOptions** request fulfills the following requirements:
 - [S4] It has HTTP 200 response code AND
 - [S5] **soapenv:Body** element has child element **trt:GetVideoSourceConfigurationOptionsResponse**.

FAIL -

- The Client failed PASS criteria.

6.15.6 SET VIDEO SOURCE CONFIGURATION

Test Label: Configure Video Source Configuration - Set Video Source Configuration

Test Case ID: VIDEOSOURCECONFIGURATIONS-4

Feature Under Test: Set Video Source Configuration
(VideoSourceConfigurations_SetVideoSourceConfiguration)

Test Purpose: To verify that Client is able to change video source configuration provided by Device using the **SetVideoSourceConfiguration** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetVideoSourceConfiguration** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetVideoSourceConfiguration** request message to change video source configuration on the Device.
2. Device responds with code HTTP 200 OK and **SetVideoSourceConfigurationResponse** message.

Test Result:**PASS -**

- Client **SetVideoSourceConfiguration** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetVideoSourceConfiguration** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:SetVideoSourceConfiguration** AND
 - [S2] **trt:SetVideoSourceConfiguration/trt:Configuration/@token** element has non-empty string value AND
- Device response to the **SetVideoSourceConfiguration** request fulfills the following requirements:
 - [S3] It has HTTP 200 response code AND
 - [S4] **soapenv:Body** element has child element **trt:SetVideoSourceConfigurationResponse**.

FAIL -

- The Client failed PASS criteria.

6.15.7 GET COMPATIBLE VIDEO SOURCE CONFIGURATIONS

Test Label: Configure Video Source Configuration - Get Compatible Video Source Configurations

Test Case ID: VIDEOSOURCECONFIGURATIONS-5

Feature Under Test: Get Compatible Video Source Configurations
(VideoSourceConfigurations_GetCompatibleVideoSourceConfigurations)

Test Purpose: To verify that Client is able to retrieve list of video source configurations that are compatible with a certain media profile using the **GetCompatibleVideoSourceConfigurations** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetCompatibleVideoSourceConfigurations** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetCompatibleVideoSourceConfigurations** request message to retrieve compatible video source configurations from the Device.
2. Device responds with code HTTP 200 OK and **GetCompatibleVideoSourceConfigurationsResponse** message.

Test Result:**PASS -**

- Client **GetCompatibleVideoSourceConfigurations** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetCompatibleVideoSourceConfigurations** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:GetCompatibleVideoSourceConfigurations** AND
 - [S2] **trt:GetCompatibleVideoSourceConfigurations/trt:ProfileToken** element has non-empty string value AND
- Device response to the **GetCompatibleVideoSourceConfigurations** request fulfills the following requirements:
 - [S3] It has HTTP 200 response code AND
 - [S4] **soapenv:Body** element has child element **trt:GetCompatibleVideoSourceConfigurationsResponse**.

FAIL -

- The Client failed PASS criteria.

6.15.8 ADD VIDEO SOURCE CONFIGURATION

Test Label: Video Source Configuration - add video source configuration

Test Case ID: VIDEOSOURCECONFIGURATIONS-6

Feature **Under** **Test:** Add Video Source Configuration
(VideoSourceConfigurations_AddVideoSourceConfiguration)

Test Purpose: To verify that Client is able to add a new or replace an existing video source configuration on Device by using the AddVideoSourceConfiguration operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with AddVideoSourceConfiguration operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes AddVideoSourceConfiguration request message with specified ProfileToken and ConfigurationToken elements to add an existing video source configuration to an existing media profile.
2. Device responds with code HTTP 200 OK and AddVideoSourceConfigurationResponse message.

Test Result:

PASS -

- Client **AddVideoSourceConfiguration** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **AddVideoSourceConfiguration** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<AddVideoSourceConfiguration>" tag after the "<Body>" tag AND
 - [S2] "<AddVideoSourceConfiguration>" includes tag: "<ProfileToken>" with non-empty string value of specific profile token AND
 - [S3] "<AddVideoSourceConfiguration>" includes tag: "<ConfigurationToken>" with non-empty string value of specific configuration token AND
 - [S4] Device response contains "HTTP/* 200 OK" AND
 - [S5] Device response contains "<AddVideoSourceConfigurationResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.16 Audio Streaming Test Cases

6.16.1 Feature Level Requirement:

Validated Feature: Audio Streaming (AudioStreaming)

Check Condition based on Device Features: Real Time Streaming (Media Service) and Audio (Media Service) are supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

6.16.2 Expected Scenarios Under Test:

1. Client connects to Device to configure a media profile and initiate Audio Streaming with G.711 encoding type.
2. Client is considered as supporting Audio Streaming if the following conditions are met:
 - Client is able to configure a media profile for audio streaming using the GetCompatibleAudioSourceConfigurations, AddAudioSourceConfiguration, GetCompatibleAudioEncoderConfigurations and AddAudioEncoderConfiguration operations AND
 - Client is able to initiate and retrieve audio stream with G.711 encoding type AND
 - When Device and Client support G.726 encoding type for Audio Streaming:
 - Client is able to initiate and retrieve audio stream with G.726 encoding type AND
 - When Device and Client support AAC encoding type for Audio Streaming:
 - Client is able to initiate and retrieve audio stream with AAC encoding type.
3. Client is considered as NOT supporting Audio Streaming if ANY of the following is TRUE:
 - No Valid Device Response to GetCompatibleAudioSourceConfigurations request OR
 - No Valid Device Response to AddAudioSourceConfiguration request OR
 - No Valid Device Response to GetCompatibleAudioEncoderConfigurations request OR
 - No Valid Device Response to AddAudioEncoderConfiguration request OR
 - G.711 Audio Streaming attempts detected have failed OR

- When Device and Client support G.726 encoding type for Audio Streaming:
 - Client is unable to initiate and retrieve audio stream with G.726 encoding type OR
- When Device and Client support AAC encoding type for Audio Streaming:
 - Client is unable to initiate and retrieve audio stream with AAC encoding type.

6.16.3 CONFIGURE MEDIA PROFILE FOR AUDIO STREAMING

Test Label: Audio Streaming - Configure Media Profile

Test Case ID: AUDIOSTREAMING-1

Feature Under Test: Audio Streaming - Configure Media Profile
(AudioStreaming_AudioStreamingConfigureMediaProfile)

Test Purpose: To verify that Client is able to to configure a media profile for audio streaming using the GetCompatibleAudioSourceConfigurations, AddAudioSourceConfiguration, GetCompatibleAudioEncoderConfigurations and AddAudioEncoderConfiguration operations.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with GetCompatibleAudioSourceConfigurations, AddAudioSourceConfiguration, GetCompatibleAudioEncoderConfigurations and AddAudioEncoderConfiguration operations present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetCompatibleAudioSourceConfigurations request message to retrieve all audio source configurations of Device that are compatible with a certain media profile.
2. Device responds with code HTTP 200 OK and GetCompatibleAudioSourceConfigurationsResponse message.
3. Client invokes AddAudioSourceConfiguration request message to add audio source configuration to an existing media profile.
4. Device responds with code HTTP 200 OK and AddAudioSourceConfigurationResponse message.
5. Client invokes GetCompatibleAudioEncoderConfigurations request message to retrieve all audio encoder configurations of the device that are compatible with a certain media profile.

6. Device responds with code HTTP 200 OK and GetCompatibleAudioEncoderConfigurationsResponse message.
7. Client invokes AddAudioEncoderConfiguration request message to add audio encoder configuration to an existing media profile.
8. Device responds with code HTTP 200 OK and AddAudioEncoderConfigurationResponse message.

Test Result:**PASS -**

- Client GetCompatibleAudioSourceConfigurations request message is a well-formed SOAP request (refer to onvif.xsd) AND
- Client GetCompatibleAudioSourceConfigurations request message has a proper hierarchy (refer to media.wsdl) AND
 - [S1] Client request contains "<GetCompatibleAudioSourceConfigurations>" tag after the "<Body>" tag AND
 - [S2] "<GetCompatibleAudioSourceConfigurations>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
 - [S3] Device response contains "HTTP/* 200 OK" AND
 - [S4] Device response contains "<GetCompatibleAudioSourceConfigurationsResponse>" tag AND
- Client AddAudioSourceConfiguration request message is a well-formed SOAP request (refer to onvif.xsd) AND
- Client AddAudioSourceConfiguration request message has a proper hierarchy (refer to media.wsdl) AND
 - [S5] Client request contains "<AddAudioSourceConfiguration>" tag after the "<Body>" tag AND
 - [S6] "<AddAudioSourceConfiguration>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
 - [S7] "<AddAudioSourceConfiguration>" includes tag: "<ConfigurationToken>" with non-empty string value of specific token AND
 - [S8] Device response contains "HTTP/* 200 OK" AND

- [S9] Device response contains "<AddAudioSourceConfigurationResponse>" tag AND
- Client GetCompatibleAudioEncoderConfigurations request message is a well-formed SOAP request (refer to onvif.xsd) AND
- Client GetCompatibleAudioEncoderConfigurations request message has a proper hierarchy (refer to media.wsdl) AND
 - [S10] Client request contains "<GetCompatibleAudioEncoderConfigurations>" tag after the "<Body>" tag AND
 - [S12] "<GetCompatibleAudioEncoderConfigurations>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
 - [S13] Device response contains "HTTP/* 200 OK" AND
 - [S14] Device response contains "<GetCompatibleAudioEncoderConfigurationsResponse>" tag AND
- Client AddAudioEncoderConfiguration request message is a well-formed SOAP request (refer to onvif.xsd) AND
- Client AddAudioEncoderConfiguration request message has a proper hierarchy (refer to media.wsdl) AND
 - [S15] Client request contains "<AddAudioEncoderConfiguration>" tag after the "<Body>" tag AND
 - [S16] "<AddAudioEncoderConfiguration>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
 - [S17] "<AddAudioEncoderConfiguration>" includes tag: "<ConfigurationToken>" with non-empty string value of specific token AND
 - [S18] Device response contains "HTTP/* 200 OK" AND
 - [S19] Device response contains "<AddAudioEncoderConfigurationResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.16.4 G.711 AUDIO STREAMING

Test Label: Audio Streaming - G.711

Test Case ID: AUDIOSTREAMING-2**Feature Under Test:** Audio Streaming - G.711 (AudioStreaming_AudioStreamingG711)**Test Purpose:** To verify that the Client is able to initiate and retrieve audio stream with G.711 encoding type.**Pre-Requisite:**

- The Network Trace Capture files contains at least one conversation between Client and Device with Audio Streaming of G.711 encoding type.
- Device supports G.711 encoding for Audio streaming.
- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetStreamUri** for Media Service.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetStreamUri** request message for media profile that contains Audio Source Configuration and Audio Encoder Configuration with G711 Encoding value. GetStreamUri request is set for RTP-Unicast/UDP OR RTP-Multicast/UDP OR RTP/RTSP/TCP OR RTP-Unicast/RTSP/HTTP/TCP transport.
2. Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
3. Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
4. Device responds with code RTSP 200 OK and SDP information with Media Type: "audio" and with encoding name "PCMU" or with payload type number "0".
5. Client invokes **RTSP SETUP** request without "onvif-replay" Require header and with transport parameter element to set media session parameters for G711 audio streaming.
6. Device responds with code RTSP 200 OK.
7. Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
8. Device responds with code RTSP 200 OK.
9. Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.
10. If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:**Note:** If no **GetStreamUri** (Media Service) corresponding to detected RTSP session found, the test will be assumed as NOT DETECTED.

Note: RTSP requests and RTSP response could be tunneled in HTTP if RTP-Unicast/RTSP/HTTP/TCP transport is used.

PASS -

- There is Client **RTSP DESCRIBE** request in Test Procedure
- Device response on the **RTSP DESCRIBE** request fulfills the following requirements:
 - [S1] It has RTSP 200 response code AND
 - [S2] IF SDP packet contains media type "audio" (m=audio) without session attribute "sendonly" (a=sendonly) AND with sessions attribute "rtpmap" THEN encoding name is "PCMU"
 - [S3] ELSE IF SDP packet contains media type "audio" (m=audio) without session attribute "sendonly" (a=sendonly) AND without sessions attribute "rtpmap" THEN payload type number is "0" (see [RFC 3551]) AND
- There is Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
 - [S4] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S5] It invoked after the Client **RTSP DESCRIBE** request AND
 - [S6] RTSP address that was used to send **RTSP SETUP** is correspond to corresponding media Control URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S7] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S8] It has RTSP 200 response code AND
- There is a Device response on the **GetStreamUri** request in Test Procedure fulfills the following requirements:
 - [S9] It has HTTP 200 response code AND
 - [S10] It received for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S11] It received before the Client **RTSP DESCRIBE** request AND
 - [S12] It contains **trt:MediaUri\trt:Uri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request AND
- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:

- [S13] It invoked for the same Device as for the Client **RTSP SETUP** request AND
- [S14] It invoked after the Client **RTSP SETUP** request AND
- [S15] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- [S16] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:
 - [S17] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
 - [S18] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S19] It invoked after the Client **RTSP PLAY** request AND
 - [S20] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:
 - [S21] It has RTSP 200 response code.

FAIL -

- The Client failed PASS criteria.

6.16.5 G.726 AUDIO STREAMING

Test Label: Audio Streaming - G.726

Test Case ID: AUDIOSTREAMING-3

Feature Under Test: Audio Streaming - G.726 (AudioStreaming_AudioStreamingG726)

Test Purpose: To verify that the Client is able to initiate and retrieve audio stream with G.726 encoding type.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with Audio Streaming of G.726 encoding type.
- Device supports G.726 encoding for Audio streaming.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetStreamUri** request message for media profile that contains Audio Source Configuration and Audio Encoder Configuration with G726 Encoding value. GetStreamUri request is set for RTP-Unicast/UDP OR RTP-Multicast/UDP OR RTP/RTSP/TCP OR RTP-Unicast/RTSP/HTTP/TCP transport.
2. Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
3. Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
4. Device responds with code RTSP 200 OK and SDP information with Media Type: "audio" and with encoding name "G726-*".
5. Client invokes **RTSP SETUP** request without "onvif-replay" Require header and with transport parameter element to to set media session parameters for G.726 audio streaming.
6. Device responds with code RTSP 200 OK.
7. Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
8. Device responds with code RTSP 200 OK.
9. Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.
10. If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:

Note: If no **GetStreamUri** (Media Service) corresponding to detected RTSP session found, the test will be assumed as NOT DETECTED.

Note: RTSP requests and RTSP response could be tunneled in HTTP if RTP-Unicast/RTSP/HTTP/TCP transport is used.

PASS -

- There is Client **RTSP DESCRIBE** request in Test Procedure
- Device response on the **RTSP DESCRIBE** request fulfills the following requirements:
 - [S1] It has RTSP 200 response code AND

- [S2] SDP packet contains media type "audio" (m=audio) without session attribute "sendonly" (a=sendonly) AND with sessions attribute "rtptime" with encoding name "G726-**" (see [RFC 3551]) AND
- There is Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
 - [S3] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S4] It invoked after the Client **RTSP DESCRIBE** request AND
 - [S5] RTSP address that was used to send **RTSP SETUP** is correspond to corresponding media Control URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S6] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S7] It has RTSP 200 response code AND
- There is a Device response on the **GetStreamUri** request in Test Procedure fulfills the following requirements:
 - [S8] It has HTTP 200 response code AND
 - [S9] It received for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S10] It received before the Client **RTSP DESCRIBE** request AND
 - [S11] It contains **trt:MediaUri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request AND
- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:
 - [S12] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S13] It invoked after the Client **RTSP SETUP** request AND
 - [S14] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S16] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:

- [S17] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
 - [S18] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S19] It invoked after the Client **RTSP PLAY** request AND
 - [S20] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:
 - [S21] It has RTSP 200 response code.

FAIL -

- The Client failed PASS criteria.

6.16.6 AAC AUDIO STREAMING

Test Label: Audio Streaming - AAC

Test Case ID: AUDIOSTREAMING-4

Feature Under Test: Audio Streaming - AAC (AudioStreaming_AudioStreamingAAC)

Test Purpose: To verify that the Client is able to initiate and retrieve audio stream with AAC encoding type.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with Audio Streaming of AAC encoding type.
- Device supports AAC encoding for Audio streaming.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetStreamUri** request message for media profile that contains Audio Source Configuration and Audio Encoder Configuration with AAC Encoding value. GetStreamUri request is set for RTP-Unicast/UDP OR RTP-Multicast/UDP OR RTP/RTSP/TCP OR RTP-Unicast/RTSP/HTTP/TCP transport.

2. Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
3. Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
4. Device responds with code RTSP 200 OK and SDP information with Media Type: "audio" and with encoding name "MPEG4-GENERIC" or "MP4A-LATM".
5. Client invokes **RTSP SETUP** request without "onvif-replay" Require header and with transport parameter element to set media session parameters for AAC audio streaming.
6. Device responds with code RTSP 200 OK.
7. Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
8. Device responds with code RTSP 200 OK.
9. Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.
10. If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:

Note: If no **GetStreamUri** (Media Service) corresponding to detected RTSP session found, the test will be assumed as NOT DETECTED.

Note: RTSP requests and RTSP response could be tunneled in HTTP if RTP-Unicast/RTSP/HTTP/TCP transport is used.

PASS -

- There is Client **RTSP DESCRIBE** request in Test Procedure
- Device response on the **RTSP DESCRIBE** request fulfills the following requirements:
 - [S1] It has RTSP 200 response code AND
 - [S2] SDP packet contains media type "audio" (m=audio) without session attribute "sendonly" (a=sendonly) AND with sessions attribute "rtmap" with encoding name "MPEG4-GENERIC" or "MP4A-LATM" (see [RFC 3640]) AND
- There is Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
 - [S3] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S4] It invoked after the Client **RTSP DESCRIBE** request AND
 - [S5] RTSP address that was used to send **RTSP SETUP** is correspond to corresponding media Control URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND

- [S6] It does not contain **Require** request header field with value is equal to "onvif-replay"
AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S7] It has RTSP 200 response code AND
- There is a Device response on the **GetStreamUri** request in Test Procedure fulfills the following requirements:
 - [S8] It has HTTP 200 response code AND
 - [S9] It received for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S10] It received before the Client **RTSP DESCRIBE** request AND
 - [S11] It contains **trt:MediaUri\trt:Uri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request AND
- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:
 - [S12] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S13] It invoked after the Client **RTSP SETUP** request AND
 - [S14] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S15] It does not contain **Require** request header field with value is equal to "onvif-replay"
AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:
 - [S16] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
 - [S17] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S18] It invoked after the Client **RTSP PLAY** request AND
 - [S19] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND

- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:
 - [S20] It has RTSP 200 response code.

FAIL -

- The Client failed PASS criteria.

6.17 Metadata Configurations Test Cases

6.17.1 Feature Level Requirement:

Validated Feature: Metadata Configurations (MetadataConfigurations)

Check Condition based on Device Features: Media Service is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

6.17.2 Expected Scenarios Under Test:

1. Client connects to Device to list and modify Metadata Configurations.
2. Client is considered as supporting Metadata Configurations if the following conditions are met:
 - Client is able to get metadata parameter options by using **GetMetadataConfigurationOptions** operation AND
 - Client is able to modify an existing metadata configuration by using **SetMetadataConfiguration** command.
3. Client is considered as NOT supporting Metadata Configurations if ANY of the following is TRUE:
 - No Valid Device Response to **GetMetadataConfigurations** request if detected OR
 - No Valid Device Response to **GetMetadataConfiguration** request if detected OR
 - No Valid Device Response to **GetMetadataConfigurationOptions** request OR
 - No Valid Device Response to **SetMetadataConfiguration** request.

6.17.3 LIST METADATA CONFIGURATIONS

Test Label: Metadata Configurations - List All Existing Metadata Configurations

Test Case ID: METADATACONFIGURATIONS-1

Feature Under Test: List Metadata Configurations
(MetadataConfigurations_GetMetadataConfigurations)

Test Purpose: To verify that list of all existing metadata configurations from Device is received by Client using the **GetMetadataConfigurations** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetMetadataConfigurations** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetMetadataConfigurations** request message to retrieve complete list of available metadata configurations from Device.
2. Device responds with code HTTP 200 OK and **GetMetadataConfigurationsResponse** message.

Test Result:

PASS -

- Client **GetMetadataConfigurations** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetMetadataConfigurations** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:GetMetadataConfigurations** AND
- Device response on the **GetMetadataConfigurations** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **trt:GetMetadataConfigurationsResponse**.

FAIL -

- The Client failed PASS criteria.

6.17.4 GET SPECIFIC METADATA CONFIGURATION

Test Label: Metadata Configurations - Gets a Specific Metadata Configuration

Test Case ID: METADATACONFIGURATIONS-2

Feature Under Test: Get Specific Metadata Configuration
(MetadataConfigurations_GetMetadataConfiguration)

Test Purpose: To verify that Client is able to retrieve a specific metadata configuration from Device by using the **GetMetadataConfiguration** operation.

Pre-Requirement:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetMetadataConfiguration** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetMetadataConfiguration** request message with specified ConfigurationToken.
2. Device responds with code HTTP 200 OK and **GetMetadataConfigurationResponse**.

Test Result:

PASS -

- Client **GetMetadataConfiguration** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetMetadataConfiguration** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:GetMetadataConfiguration** AND
 - [S2] **trc:GetMetadataConfiguration/trt:ConfigurationToken** element has non-empty string value of specific metadata configuraton token AND
- Device response on the **GetMetadataConfiguration** request fulfills the following requirements:
 - [S3] It has HTTP 200 response code AND
 - [S4] **soapenv:Body** element has child element **trt:GetMetadataConfigurationResponse**.

FAIL -

- The Client failed PASS criteria.

6.17.5 GET METADATA CONFIGURATION OPTIONS

Test Label: Metadata Configurations - Get Metadata Configuration Options

Test Case ID: METADATACONFIGURATIONS-3

Feature Under Test: Get Metadata Configuration Options
(MetadataConfigurations_GetMetadataConfigurationOptions)

Test Purpose: To verify that Client is able to get metadata configuration options by using **GetMetadataConfigurationOptions** operation

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetMetadataConfigurationOptions** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetMetadataConfigurationOptions** request message to retrieve supported metadata configuration options from Device.
2. Device responds with code HTTP 200 OK and **GetMetadataConfigurationOptionsResponse** message.

Test Result:

PASS -

- Client **GetMetadataConfigurationOptions** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetMetadataConfigurationOptions** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:GetMetadataConfigurationOptions** AND
 - If it contains **trt:GetMetadataConfigurationOptions/trt:ConfigurationToken** element then it fulfills the following requirements (else skip the check):
 - [S2] **trc:GetMetadataConfigurationOptions/trt:ConfigurationToken** element has non-empty string value of specific metadata configuraton token AND

- If it contains **trt:GetMetadataConfigurationOptions/trt:ProfileToken** element then it fulfills the following requirements (else skip the check):
 - [S3] **trc:GetMetadataConfigurationOptions/trt:ProfileToken** element has non-empty string value of specific profile token AND
- Device response on the **GetMetadataConfigurationOptions** request fulfills the following requirements:
 - [S4] It has HTTP 200 response code AND
 - [S5] **soapenv:Body** element has child element **trt:GetMetadataConfigurationOptionsResponse**.

FAIL -

- The Client failed PASS criteria.

6.17.6 MODIFY METADATA CONFIGURATION

Test Label: Metadata Configurations - Modify Metadata Configuration

Test Case ID: METADATACONFIGURATIONS-4

Feature Under Test: Modify Metadata Configuration
(MetadataConfigurations_ModifyMetadataConfiguration)

Test Purpose: To verify that Client is able to modify metadata configuration on Device by using the **SetMetadataConfiguration** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetMetadataConfiguration** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetMetadataConfiguration** request message to change metadata configuration settings with any modifications of parameters.
2. Device responds with code HTTP 200 OK and **SetMetadataConfigurationResponse** message.

Test Result:

PASS -

- Client **SetMetadataConfiguration** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetMetadataConfiguration** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **trt:SetMetadataConfiguration** AND
 - [S2] **trt:SetMetadataConfiguration/trt:Configuration/@token** attribute has non-empty string value of specific configuration token AND
- Device response on the **SetMetadataConfiguration** request fulfills the following requirements:
 - [S3] It has HTTP 200 response code AND
 - [S4] **soapenv:Body** element has child element **trt:SetMetadataConfigurationResponse**.

FAIL -

- The Client failed PASS criteria.

6.18 Multiple Audio Sources Test Cases

6.18.1 Feature Level Requirement:

Validated Feature: Multiple Audio Sources (MultipleAudioSources)

Check Condition based on Device Features: Real Time Streaming (Media Service) and Audio (Media Service) are supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

6.18.2 Expected Scenarios Under Test:

1. Client connects to Device to get all Audio Sources.
2. Client obtains audio streaming for each Audio Source provided by a Device.
3. Client is considered as supporting Multiple Audio Sources if the following conditions are met:
 - Client is able to get profile list by using **GetProfiles** operation AND

- Client is able to to initiate and retrieve audio stream for each Audio Source provided by a Device using **GetStreamUri** command and **RTSP** commands.
4. Client is considered as NOT supporting Multiple Audio Sources if ANY of the following is TRUE:
- No Valid Device Response to **GetProfiles** request OR
 - Client is unable to initiate and retrieve audio streaming for at least one Audio Source provided by a Device.

6.18.3 STREAMING WITH ALL AUDIO SOURCES DETECTED IN GET PROFILES

Test Label: Multiple Audio Sources - Streaming with all Audio Sources detected in GetProfilesResponse

Test Case ID: MULTIPLEAUDIOSOURCES-1

Feature Under Test: Streaming For Audio Sources From GetProfiles (MultipleAudioSources_StreamingForAudioSourcesFromGetProfiles)

Test Purpose: To verify that Client is able to obtaine audio streaming for each audio source provided by a Device in GetProfiles responses.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with audio streaming present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetProfiles** request messages to retrieve complete list of available media profiles with audio source configurations from Device.
2. Device responds with code HTTP 200 OK and **GetProfilesResponse** message.
3. Client initiate audio streaming for each Audio Source token detected in GetProfilesResponse:
 - Client selects existing media profile with required Audio Source token or modifies media profile to have required Audio Source token or creates media profile with required Audio Source token.
 - Client invokes **GetStreamUri** request for this media profile.

- Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
- Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
- Device responds with code RTSP 200 OK and SDP information with Media Type: "audio".
- Client invokes **RTSP SETUP** request without "onvif-replay" Require header and with transport parameter element to set media session parameters for audio streaming.
- Device responds with code RTSP 200 OK.
- Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
- Device responds with code RTSP 200 OK.
- Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.
- If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:

Note: If Client does not initiate audio streaming then Test shall be deemed as "NOT DETECTED".

PASS -

- For each Audio Source Token listed by HelperGetAudioSourcesListFromGetProfiles (see [Annex A.4](#)) there is a audio stream in Test Procedure that fulfills the following requirements:
 - There is a Client **GetStreamUri** request that fulfills the following requirements:
 - [S1] It invoked for the media profile which contains Audio Source Configuration with this Audio Source Token (see [Annex A.5](#) HelperGetAudioSourceTokenUsedForStreaming to get audio source token from media profile) AND
 - Device response on the **GetStreamUri** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **trt:GetStreamUriResponse** AND
 - There is a RTSP session in Test Procedure that fulfills the following requirements:
 - [S5] It invoked for the uri from **GetStreamUri** response AND
 - [S6] It started audio streaming according to HelperFindAudioStreamingForGetStreamUri (see [Annex A.6](#))

FAIL -

- The Client failed PASS criteria.

6.19 PTZ - Listing Test Cases

6.19.1 Feature Level Requirement:

Validated Feature: PTZ Listing (PtzListing)

Check Condition based on Device Features: PTZ Service is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

Profile T Requirement: Conditional

6.19.2 Expected Scenarios Under Test:

1. Client connects to Device to read PTZ capabilities.
2. Client is considered as supporting PTZ - Listing if the following conditions are met:
 - Client is able to read PTZ capabilities from PTZ Node using EITHER GetNodes OR GetNode operations.
3. Client is considered as NOT supporting PTZ - Listing if ANY of the following is TRUE:
 - No Valid Device Response to GetNodes request AND
 - No Valid Device Response to GetNode request.

6.19.3 GET NODES

Test Label: PTZ Listing - GetNodes

Test Case ID: PTZLISTING-1

Feature Under Test: Get Nodes (PtzListing_GetNodes)

Test Purpose: To verify that list of all existing PTZ capabilities from Device is received by Client using the GetNodes operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with GetNodes operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetNodes request message to retrieve complete PTZ capabilities list from Device.
2. Device responds with code HTTP 200 OK and GetNodesResponse message.

Test Result:**PASS -**

- Client **GetNodes** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetNodes** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetNodes>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND
 - [S3] Device response contains "<GetNodesResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.19.4 GET NODE

Test Label: PTZ Listing - GetNode

Test Case ID: PTZLISTING-2

Feature Under Test: Get Node (PtzListing_GetNode)

Test Purpose: To verify that Client is able to retrieve a specific PTZ capability properties from Device using the GetNode operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with GetNode operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetNode request message to retrieve a specific PTZ capability properties from Device.

2. Device responds with code HTTP 200 OK and GetNodeResponse message.

Test Result:**PASS -**

- Client **GetNode** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetNode** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetNode>" tag after the "<Body>" tag AND
 - [S2] "<GetNode>" includes tag: "<NodeToken>" with non-empty string value of specific token AND
 - [S3] Device response contains "HTTP/* 200 OK" AND
 - [S4] Device response contains "<GetNodeResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.20 PTZ - Configuration Test Cases

6.20.1 Feature Level Requirement:

Validated Feature: PTZ Configuration (PtzConfiguration)

Check Condition based on Device Features: PTZ Service and Media Service are supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

6.20.2 Expected Scenarios Under Test:

1. Client connects to Device to add PTZ configuration to a media profile.
2. Client is considered as supporting PTZ - Configuration if the following conditions are met:
 - Client is able to add PTZ configuration to an existing media profile using GetConfigurations operation AND AddPTZConfiguration operation.

3. Client is considered as NOT supporting PTZ - Configuration if ANY of the following is TRUE:
 - No Valid Device Response to GetConfigurations request OR
 - No Valid Device Response to AddPTZConfiguration request.

6.20.3 ADD PTZ CONFIGURATION

Test Label: PTZ Configuration - Add PTZ Configuration

Test Case ID: PTZCONFIGURATION-1

Feature Under Test: Add PTZ Configuration to Media Profile
(PtzConfiguration_AddPtzConfiguration)

Test Purpose: To verify that Client is able to add PTZ configuration to a profile using GetConfigurations and AddPTZConfiguration operations.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with GetConfigurations and AddPTZConfiguration operations present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetConfigurations request message to retrieve all available PTZ configurations from Device.
2. Device responds with code HTTP 200 OK and GetConfigurationsResponse message.
3. Client invokes AddPTZConfiguration request message to add a PTZ configuration to an existing media profile.
4. Device responds with code HTTP 200 OK and AddPTZConfigurationResponse message.

Test Result:

PASS -

- Client **GetConfigurations** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetConfigurations** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetConfigurations>" tag after the "<Body>" tag AND
 - [S2] Device response contains "HTTP/* 200 OK" AND

- [S3] Device response contains "<GetConfigurationsResponse>" tag AND
- Client **AddPTZConfiguration** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **AddPTZConfiguration** request in Test Procedure fulfills the following requirements:
 - [S4] Client request contains "<AddPTZConfiguration>" tag after the "<Body>" tag AND
 - [S5] "<AddPTZConfiguration>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
 - [S6] "<AddPTZConfiguration>" includes tag: "<ConfigurationToken>" with non-empty string value of specific token AND
 - [S7] Device response contains "HTTP/* 200 OK" AND
 - [S8] Device response contains "<AddPTZConfigurationResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.21 PTZ Pan Tilt Continuous Positioning Test Cases

6.21.1 Feature Level Requirement:

Validated Feature: Continuous Move (PtzPanTiltContinuousPositioning)

Check Condition based on Device Features: PTZ Continuous Pan Tilt movement is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

Profile T Requirement: Mandatory

6.21.2 Expected Scenarios Under Test:

1. Client connects to Device to control PTZ Pan Tilt position using continuous move.
2. Client is considered as supporting PTZ Pan Tilt Continuous Positioning if the following conditions are met:

- Client is able to move PTZ Device using the ContinuousMove operation with specified PanTilt element AND
 - Client is able to stop PTZ Pan Tilt Device movement using the Stop operation OR using ContinuousMove operation with zero values in PanTilt element.
3. Client is considered as NOT supporting PTZ Pan Tilt Continuous Positioning if ANY of the following is TRUE:
- Client is unable to move a PTZ device using the ContinuousMove operation with specified PanTilt element OR
 - Client is unable to stop PTZ Pan Tilt movement using EITHER Stop operation OR using ContinuousMove operation OR
 - No Valid Device Response to **Stop** request if detected OR
 - No Valid Device Response to **ContinuousMove** request with zero "x" and "y" attributes values in PanTilt element if detected.

6.21.3 PTZ CONTINUOUS MOVE PAN/TILT

Test Label: PTZ Continuous Positioning - ContinuousMove PanTilt

Test Case ID: PTZPANTILTCONTINUOUSPOSITIONING-1

Feature	Under	Test:	Pan	Tilt	Continuous	Move
(PtzPanTiltContinuousPositioning_ContinuousMovePanTilt)						

Test Purpose: To verify that Client is able to move a PTZ Device using the ContinuousMove operation with specified PanTilt element.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with ContinuousMove operation present.
- Device supports PTZContinuousPanTilt.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes ContinuousMove request message to start move of PTZ Device using specific value of PanTilt element.
2. Device responds with code HTTP 200 OK and ContinuousMoveResponse message.

Test Result:**PASS -**

- Client **ContinuousMove** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **ContinuousMove** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<ContinuousMove>" tag after the "<Body>" tag AND
 - [S2] "<ContinuousMove>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
 - [S4] "<Velocity>" includes tag: "<PanTilt>" AND
 - [S7] Device response contains "HTTP/* 200 OK" AND
 - [S8] Device response contains "<ContinuousMoveResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.21.4 PTZ PAN TILT STOP

Test Label: PTZ Pan Tilt Continuous Positioning - Stop

Test Case ID: PTZPANTILTCONTINUOUSPOSITIONING-2

Feature Under Test: Stop Pan Tilt Movement (PtzPanTiltContinuousPositioning_PanTiltStop)

Test Purpose: To verify that Client is able to stop a PTZ Pan Tilt Device movement using the Stop operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with Stop operation with skipped PanTilt element or with PanTilt = true present

Test Procedure (expected to be reflected in network trace file):

1. Client invokes Stop request message to stop ongoing pan tilt movements of PTZ Device.
2. Device responds with code HTTP 200 OK and StopResponse message.

Test Result:**PASS -**

- Client **Stop** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **Stop** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tptz:Stop** AND
 - [S2] **tptz:Stop/tptz:ProfileToken** element has non-empty string value of specific token AND
 - [S3] If **tptz:Stop** contains **tptz:PanTilt** element then **tptz:Stop/tptz:PanTilt** = true AND
- Device response on the **Stop** request fulfills the following requirements:
 - [S4] It has HTTP 200 response code AND
 - [S5] **soapenv:Body** element has child element **tptz:StopResponse**.

FAIL -

- The Client failed PASS criteria.

6.21.5 STOP PAN TILT MOVEMENT USING PTZ CONTINUOUS MOVE

Test Label: PTZ Continuous Positioning - Stop Pan Tilt Movement using ContinuousMove

Test Case ID: PTZPANTILTCONTINUOUSPOSITIONING-3

Feature Under Test: Stop Pan Tilt Movement using Continuous Move (PtzPanTiltContinuousPositioning_PanTiltStopUsingPTZContinuousMove)

Test Purpose: To verify that Client is able to stop a PTZ Pan Tilt Device movement using ContinuousMove operation with zero values in PanTilt element.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with ContinuousMove operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes ContinuousMove request message with zero "x" and "y" attributes values in PanTilt element.
2. Device responds with code HTTP 200 OK and ContinuousMoveResponse message.

Test Result:

NOTE: In case Client does not send ContinuousMove request message with zero "x" and "y" attributes values in PanTilt element then the test shall be deemed as "NOT DETECTED".

PASS -

- There is client **ContinuousMove** request messages which corresponds to the following requirements (else skip the check):
 - Client **ContinuousMove** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
 - Client **ContinuousMove** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tptz:ContinuousMove** AND
 - [S2] **tptz:ContinuousMove/tptz:ProfileToken** element has non-empty string value of specific token AND
 - [S3] **tptz:ContinuousMove/tptz:Velocity** contain tag **tt:PanTilt** AND
 - [S4] **tptz:ContinuousMove/tptz:Velocity/tt:PanTilt/@x** attribute value is equal to 0 AND
 - [S5] **tptz:ContinuousMove/tptz:Velocity/tt:PanTilt/@y** attribute value is equal to 0 AND
 - Device response on the **ContinuousMove** request fulfills the following requirements:
 - [S6] It has HTTP 200 response code AND
 - [S7] **soapenv:Body** element has child element **tptz:ContinuousMoveResponse**.

FAIL -

- The Client failed PASS criteria.

6.22 PTZ Zoom Continuous Positioning Test Cases

6.22.1 Feature Level Requirement:

Validated Feature: Zoom Continuous Move (PtzZoomContinuousPositioning)

Check Condition based on Device Features: PTZ Continuous Zoom movement is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

Profile T Requirement: Mandatory

6.22.2 Expected Scenarios Under Test:

1. Client connects to Device to control PTZ Zoom position using continuous move.
2. Client is considered as supporting PTZ Zoom Continuous Positioning if the following conditions are met:
 - Client is able to move PTZ Device using the ContinuousMove operation with specified Zoom element AND
 - Client is able to stop PTZ Zoom Device movement using the Stop operation OR using ContinuousMove operation with zero values in Zoom element.
3. Client is considered as NOT supporting PTZ Zoom Continuous Positioning if ANY of the following is TRUE:
 - Client is unable to move a PTZ device using the ContinuousMove operation with specified Zoom element OR
 - Client is unable to stop PTZ Zoom movement using EITHER Stop operation OR using ContinuousMove operation OR
 - No Valid Device Response to **Stop** request if detected OR
 - No Valid Device Response to **ContinuousMove** request with zero "x" attributes values in Zoom element if detected.

6.22.3 PTZ CONTINUOUS MOVE ZOOM

Test Label: PTZ Continuous Positioning - ContinuousMove Zoom

Test Case ID: PTZZOOMCONTINUOUSPOSITIONING-1

Feature	Under	Test:	Zoom	Continuous	Move
(PtzZoomContinuousPositioning_ContinuousMoveZoom)					

Test Purpose: To verify that Client is able to change zoom of PTZ Device using the ContinuousMove operation with specified Zoom element.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with ContinuousMove operation present.
- Device supports PTZContinuousZoom.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes ContinuousMove request message to change zoom of PTZ Device using specific value of Zoom element.
2. Device responds with code HTTP 200 OK and ContinuousMoveResponse message.

Test Result:**PASS -**

- Client **ContinuousMove** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **ContinuousMove** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<ContinuousMove>" tag after the "<Body>" tag AND
 - [S2] "<ContinuousMove>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
 - [S4] "<Velocity>" includes tag: "<Zoom>" AND
 - [S6] Device response contains "HTTP/* 200 OK" AND
 - [S7] Device response contains "<ContinuousMoveResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.22.4 PTZ ZOOM STOP

Test Label: PTZ Zoom Continuous Positioning - Stop

Test Case ID: PTZZOOMCONTINUOUSPOSITIONING-2

Feature Under Test: Stop Zoom Movement (PtzZoomContinuousPositioning_ZoomStop)

Test Purpose: To verify that Client is able to stop a PTZ Zoom Device movement using the Stop operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with Stop operation with skipped Zoom element or with Zoom = true present

Test Procedure (expected to be reflected in network trace file):

1. Client invokes Stop request message to stop ongoing zoom movements of PTZ Device.
2. Device responds with code HTTP 200 OK and StopResponse message.

Test Result:**PASS -**

- Client **Stop** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **Stop** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tptz:Stop** AND
 - [S2] **tptz:Stop/tptz:ProfileToken** element has non-empty string value of specific token AND
 - [S3] If **tptz:Stop** contains **tptz:Zoom** element then **tptz:Stop/tptz:Zoom** = true AND
- Device response on the **Stop** request fulfills the following requirements:
 - [S4] It has HTTP 200 response code AND
 - [S5] **soapenv:Body** element has child element **tptz:StopResponse**.

FAIL -

- The Client failed PASS criteria.

6.22.5 STOP ZOOM MOVEMENT USING PTZ CONTINUOUS MOVE

Test Label: PTZ Continuous Positioning - Stop Zoom Movement using ContinuousMove

Test Case ID: PTZZOOMCONTINUOUSPOSITIONING-3

Feature Under Test: Stop Zoom Movement using Continuous Move
(PtzZoomContinuousPositioning_ZoomStopUsingPTZContinuousMove)

Test Purpose: To verify that Client is able to stop a PTZ Zoom Device movement using ContinuousMove operation with zero values in Zoom element.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with ContinuousMove operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes ContinuousMove request message with zero "x" attribute value in Zoom element.
2. Device responds with code HTTP 200 OK and ContinuousMoveResponse message.

Test Result:

NOTE: In case Client does not send ContinuousMove request message with zero "x" attribute value in Zoom element if device supports PTZContinuousZoom then the test shall be deemed as "NOT DETECTED".

PASS -

- There is client **ContinuousMove** request messages which corresponds to the following requirements (else skip the check):
 - Client **ContinuousMove** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
 - Client **ContinuousMove** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tptz:ContinuousMove** AND
 - [S2] **tptz:ContinuousMove/tptz:ProfileToken** element has non-empty string value of specific token AND
 - [S3] **tptz:ContinuousMove/tptz:Velocity** contain tag **tt:Zoom** AND
 - [S4] **tptz:ContinuousMove/tptz:Velocity/tt:Zoom/@x** attribute value is equal to 0.
 - Device response on the **ContinuousMove** request fulfills the following requirements:
 - [S5] It has HTTP 200 response code AND
 - [S6] **soapenv:Body** element has child element **tptz:ContinuousMoveResponse**.

FAIL -

- The Client failed PASS criteria.

6.23 PTZ Pan Tilt Absolute Positioning Test Cases

6.23.1 Feature Level Requirement:

Validated Feature: PTZ Pan Tilt Absolute Positioning (PtzPanTiltAbsolutePositioning)

Check Condition based on Device Features: Pan Tilt Absolute Movement and Profile S are supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

Profile T Requirement: None

6.23.2 Expected Scenarios Under Test:

1. Client connects to Device to control the pan tilt position using absolute positioning.
2. Client is considered as supporting PTZ Pan Tilt Absolute Positioning if the following conditions are met:
 - Client is able to move PTZ Device using the AbsoluteMove operation by Move a PTZ Device using the AbsoluteMove operation with specified PanTilt element.
3. Client is considered as NOT supporting PTZ Pan Tilt Absolute Positioning if ANY of the following is TRUE:
 - No Valid Device Response to AbsoluteMove request with specified PanTilt element.

6.23.3 PTZ ABSOLUTE MOVE PAN/TILT

Test Label: PTZ Absolute Positioning - AbsoluteMove PanTilt

Test Case ID: PTZPANTILTABSOLUTEPOSITIONING-1

Feature	Under	Test:	Pan	Tilt	Absolute	Move
(PtzPanTiltAbsolutePositioning_AbsoluteMovePanTilt)						

Test Purpose: To verify that Client is able to move a PTZ Device using the AbsoluteMove operation with specified PanTilt element.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with AbsoluteMove operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes AbsoluteMove request message to move of PTZ Device using specific value of PanTilt element.
2. Device responds with code HTTP 200 OK and AbsoluteMoveResponse message.

Test Result:

NOTE: If Client AbsoluteMove request message does not contain "<PanTilt>" tag inside "<Position>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **AbsoluteMove** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **AbsoluteMove** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<AbsoluteMove>" tag after the "<Body>" tag AND
 - [S2] "<AbsoluteMove>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
 - [S4] "<Position>" includes tag: "<PanTilt>" AND
 - [S7] Device response contains "HTTP/* 200 OK" AND
 - [S8] Device response contains "<AbsoluteMoveResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.24 PTZ Zoom Absolute Positioning Test Cases

6.24.1 Feature Level Requirement:

Validated Feature: PTZ Zoom Absolute Positioning (PtzZoomAbsolutePositioning)

Check Condition based on Device Features: Zoom Absolute Movement and Profile S are supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

Profile T Requirement: None

6.24.2 Expected Scenarios Under Test:

1. Client connects to Device to control the zoom position using absolute positioning.
2. Client is considered as supporting PTZ Zoom Absolute Positioning if the following conditions are met:
 - Client is able to change zoom of PTZ Device using the AbsoluteMove operation with specified Zoom element.
3. Client is considered as NOT supporting PTZ Zoom Absolute Positioning if ANY of the following is TRUE:
 - No Valid Device Response to AbsoluteMove request with specified Zoom element.

6.24.3 PTZ ABSOLUTE MOVE ZOOM

Test Label: PTZ Absolute Positioning - AbsoluteMove Zoom

Test Case ID: PTZZOOMABSOLUTEPOSITIONING-1

Feature Under Test: Zoom Absolute Move (PtzZoomAbsolutePositioning_AbsoluteZoom)

Test Purpose: To verify that Client is able to change zoom of PTZ Device using the AbsoluteMove operation with specified Zoom element.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with AbsoluteMove operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes AbsoluteMove request message to change zoom of PTZ Device using specific value of Zoom element.
2. Device responds with code HTTP 200 OK and AbsoluteMoveResponse message.

Test Result:

PASS -

- Client **AbsoluteMove** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **AbsoluteMove** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<AbsoluteMove>" tag after the "<Body>" tag AND
 - [S2] "<AbsoluteMove>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
 - [S4] "<Position>" includes tag: "<Zoom>" AND
 - [S6] Device response contains "HTTP/* 200 OK" AND
 - [S7] Device response contains "<AbsoluteMoveResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.25 PTZ Pan Tilt Relative Positioning Test Cases

6.25.1 Feature Level Requirement:

Validated Feature: PTZ Pan Tilt Relative Positioning (PtzPanTiltRelativePositioning)

Check Condition based on Device Features: Relative Tan Tilt move and Profile S are supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

6.25.2 Expected Scenarios Under Test:

1. Client connects to Device to control the position using relative positioning.
2. Client is considered as supporting PTZ Pan Tilt Relative Positioning if the following conditions are met:
 - Client is able to move PTZ Device using the RelativeMove operation by Move a PTZ Device using the RelativeMove operation with specified PanTilt element.
3. Client is considered as NOT supporting PTZ Pan Tilt Relative Positioning if ANY of the following is TRUE:

- No Valid Device Response to RelativeMove request with specified PanTilt element.

6.25.3 PTZ RELATIVE MOVE PAN/TILT

Test Label: PTZ Relative Positioning - Relative Move PanTilt

Test Case ID: PTZPANTILTRELATIVEPOSITIONING-1

Feature	Under	Test:	Pan	Tilt	Relative	Move
(PtzPanTiltRelativePositioning_PtzRelativeMovePanTilt)						

Test Purpose: To verify that Client is able to move a PTZ Device using the RelativeMove operation with specified PanTilt element.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with RelativeMove operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes RelativeMove request message to move of PTZ Device using specific value of PanTilt element.
2. Device responds with code HTTP 200 OK and RelativeMoveResponse message.

Test Result:

NOTE: If Client RelativeMove request message does not contain "<PanTilt>" tag inside "<Translation>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **RelativeMove** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **RelativeMove** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<RelativeMove>" tag after the "<Body>" tag AND
 - [S2] "<RelativeMove>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
 - [S4] "<Translation>" includes tag: "<PanTilt>" AND
 - [S7] Device response contains "HTTP/* 200 OK" AND

- [S8] Device response contains "<RelativeMoveResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.26 PTZ Zoom Relative Positioning Test Cases

6.26.1 Feature Level Requirement:

Validated Feature: PTZ Pan Tilt Relative Positioning (PtzZoomRelativePositioning)

Check Condition based on Device Features: Relative Zoom move and Profile S are supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

6.26.2 Expected Scenarios Under Test:

1. Client connects to Device to control the position using relative positioning.
2. Client is considered as supporting PTZ Zoom Relative Positioning if the following conditions are met:
 - Client is able to change zoom of PTZ Device using the RelativeMove operation with specified Zoom element.
3. Client is considered as NOT supporting PTZ Zoom Relative Positioning if ANY of the following is TRUE:
 - No Valid Device Response to RelativeMove request with specified Zoom element.

6.26.3 PTZ RELATIVE MOVE ZOOM

Test Label: PTZ Relative Positioning - Relative Move Zoom

Test Case ID: PTZZOOMRELATIVEPOSITIONING-1

Feature Under Test: Zoom Relative Move (PtzZoomRelativePositioning_PtzRelativeMoveZoom)

Test Purpose: To verify that Client is able to change zoom of PTZ Device using the RelativeMove operation with specified Zoom element.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with RelativeMove operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes RelativeMove request message to change zoom of PTZ Device using specific value of Zoom element.
2. Device responds with code HTTP 200 OK and RelativeMoveResponse message.

Test Result:

NOTE: If Client AbsoluteMove request message does not contain "<Zoom>" tag inside "<Translation>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **RelativeMove** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **RelativeMove** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<RelativeMove>" tag after the "<Body>" tag AND
 - [S2] "<RelativeMove>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
 - [S4] "<Translation>" includes tag: "<Zoom>" AND
 - [S6] Device response contains "HTTP/* 200 OK" AND
 - [S7] Device response contains "<RelativeMoveResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.27 PTZ Presets Test Cases

6.27.1 Feature Level Requirement:

Validated Feature: PTZ Presets (PtzPresets)

Check Condition based on Device Features: PTZ Presets is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

Profile T Requirement: Conditional

6.27.2 Expected Scenarios Under Test:

1. Client connects to Device to manage the presets of a PTZ Node.
2. Client is considered as supporting PTZ Presets if the following conditions are met:
 - Client is able to list the presets using the GetPresets operation AND
 - Client is able to move a PTZ Device to a specific preset using the GotoPreset operation.
3. Client is considered as NOT supporting PTZ Presets if ANY of the following is TRUE:
 - No Valid Device Response to GetPresets request OR
 - No Valid Device Response to GotoPreset request.

6.27.3 PTZ GET PRESETS

Test Label: PTZ Presets - GetPresets

Test Case ID: PTZPRESETS-1

Feature Under Test: Get Presets (PtzPresets_PtzGetPresets)

Test Purpose: To verify that Client is able to list the presets using the GetPresets operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with GetPresets operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetPresets request message to list the available presets from Device.
2. Device responds with code HTTP 200 OK and GetPresetsResponse message.

Test Result:

PASS -

- Client **GetPresets** request messages are valid according to XML Schemas listed in [Namespaces](#) AND

- Client **GetPresets** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GetPresets>" tag after the "<Body>" tag AND
 - [S2] "<GetPresets>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
 - [S3] Device response contains "HTTP/* 200 OK" AND
 - [S4] Device response contains "<GetPresetsResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.27.4 PTZ GOTO PRESET

Test Label: PTZ Presets - GotoPreset

Test Case ID: PTZPRESETS-2

Feature Under Test: Goto Preset (PtzPresets_PtzGotoPreset)

Test Purpose: To verify that Client is able to move a PTZ Device to a specific preset using the GotoPreset operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with GotoPreset operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GotoPreset request message to move PTZ Device to specific preset.
2. Device responds with code HTTP 200 OK and GotoPresetResponse message.

Test Result:**PASS -**

- Client **GotoPreset** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GotoPreset** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GotoPreset>" tag after the "<Body>" tag AND

- [S2] "<GotoPreset>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
- [S3] "<GotoPreset>" includes tag: "<PresetToken>" with non-empty string value of specific token AND
- [S4] Device response contains "HTTP/* 200 OK" AND
- [S5] Device response contains "<GotoPresetResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.28 PTZ Home Position Test Cases

6.28.1 Feature Level Requirement:

Validated Feature: PTZ Home Position (PtzHomePosition)

Check Condition based on Device Features: PTZ Home Position is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

Profile T Requirement: Conditional

6.28.2 Expected Scenarios Under Test:

1. Client connects to Device to manage the home position of a PTZ Node.
2. Client is considered as supporting PTZ Home Position if the following conditions are met:
 - Client is able to move PTZ Device to its home position using the GotoHomePosition operation
3. Client is considered as NOT supporting PTZ Home Position if ANY of the following is TRUE:
 - No Valid Device Response to GotoHomePosition request.

6.28.3 PTZ HOME POSITION

Test Label: PTZ Presets - GotoHomePosition

Test Case ID: PTZHOMEPOSITION-1

Feature Under Test: Goto Home Position (PtzHomePosition_PtzGotoHomePosition)

Test Purpose: To verify that Client is able to move PTZ Device to its home position using the GotoHomePosition operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with GotoHomePosition operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GotoHomePosition request message to move PTZ Device to its home position.
2. Device responds with code HTTP 200 OK and GotoHomeResponse message.

Test Result:

PASS -

- Client **GotoHomePosition** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GotoHomePosition** request in Test Procedure fulfills the following requirements:
 - [S1] Client request contains "<GotoHomePosition>" tag after the "<Body>" tag AND
 - [S2] "<GotoHomePosition>" includes tag: "<ProfileToken>" with non-empty string value of specific token AND
 - [S3] Device response contains "HTTP/* 200 OK" AND
 - [S4] Device response contains "<GotoHomePositionResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.29 PTZ - Auxiliary Command Test Cases

6.29.1 Feature Level Requirement:

Validated Feature: PTZ Auxiliary Command (PtzAuxiliaryCommand)

Check Condition based on Device Features: Auxiliary Operations (PTZ Service) and Profile S are supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

6.29.2 Expected Scenarios Under Test:

1. Client connects to Device to manage the auxiliary commands of a PTZ Node.
2. Client is considered as supporting PTZ - Auxiliary Command if the following conditions are met:
 - Client is able to call an auxiliary operation on Device using the `SendAuxiliaryCommand` operation.
3. Client is considered as NOT supporting PTZ - Auxiliary Command if ANY of the following is TRUE:
 - No Valid Device Response to `SendAuxiliaryCommand` request.

6.29.3 PTZ SEND AUXILIARY COMMAND

Test Label: PTZ Auxiliary Command - Send Auxiliary Command

Test Case ID: PTZAUXILIARYCOMMAND-1

Feature	Under	Test:	Send	Auxiliary	Command
(PtzAuxiliaryCommand_PtzSendAuxiliaryCommand)					

Test Purpose: To verify that Client is able to call an auxiliary operation on Device using the `SendAuxiliaryCommand` operation (PTZ Service).

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with `SendAuxiliaryCommand` operation (PTZ Service) present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes `SendAuxiliaryCommand` request message (PTZ Service) to call an auxiliary operation on Device.
2. Device responds with code HTTP 200 OK and `SendAuxiliaryCommandResponse` message.

Test Result:**PASS -**

- Client **SendAuxiliaryCommand** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SendAuxiliaryCommand** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tptz:SendAuxiliaryCommand** AND
 - [S2] It contains **tptz:ProfileToken** element with non-empty string value AND
 - [S3] It contains **tptz:AuxiliaryData** element with non-empty string value AND
- Device response on the **SendAuxiliaryCommand** request fulfills the following requirements:
 - [S4] It has HTTP 200 response code AND
 - [S5] **soapenv:Body** element has child element **tptz:SendAuxiliaryCommandResponse**.

FAIL -

- The Client failed PASS criteria.

7 Test Cases for Profile Optional Features

7.1 Set Synchronization Point (Event Service) Test Cases

7.1.1 Feature Level Requirement:

Validated Feature: Set Synchronization Point (SetSynchronizationPoint)

Check Condition based on Device Features: Pull Point Notification OR WS-Basic Notification is supported by Device.

Required Number of Devices: 1

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile S Requirement: Optional

Profile G Requirement: Optional

Profile T Requirement: Mandatory

Profile D Requirement: Mandatory

7.1.2 Expected Scenarios Under Test:

1. Client connects to Device to synchronize property states.
2. Client is considered as supporting Set Synchronization Point (Event Service) if the following conditions are met:
 - Client is able to synchronize property states using **SetSynchronizationPoint** operation for subscriptions AND
3. Client is considered as NOT supporting Set Synchronization Point (Event Service) if the following is TRUE:
 - No valid responses for **SetSynchronizationPoint** request OR
 - **SetSynchronizationPoint** request does not contains valid **wsa:Action** header.

7.1.3 SET SYNCHRONIZATION POINT (EVENT SERVICE)

Test Label: Set Synchronization Point - Set Synchronization Point

Test Case ID: SETSYNCHRONIZATIONPOINT-1

Feature **Under** **Test:** Set Synchronization Point
(SetSynchronizationPoint_SetSynchronizationPointAction)

Test Purpose: To verify that the Client is able to use **SetSynchronizationPoint** operation for subscription.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetSynchronizationPoint** operations present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetSynchronizationPoint** message with valid **wsa:Action** header to synchronize its properties with the properties of the device.
2. Device responds with code HTTP 200 OK and **SetSynchronizationPointResponse** message.

Test Result:**PASS -**

- Client **SetSynchronizationPoint** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetSynchronizationPoint** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tev:SetSynchronizationPoint** AND
 - [S2] It contains **wsa:Action** element in header equal to "http://www.onvif.org/ver10/events/wsdl/PullPointSubscription/SetSynchronizationPointRequest" AND
- Device response on the **SetSynchronizationPoint** request fulfills the following requirements:
 - [S3] It has HTTP 200 response code AND
 - [S4] **soapenv:Body** element has child element **tev:SetSynchronizationPointResponse**

FAIL -

- The Client failed PASS criteria.

7.2 Unsubscribe Test Cases

Validated Feature: Unsubscribe (Unsubscribe)

Check Condition based on Device Features: Pull Point Notification OR WS-Basic Notification is supported by Device.

Required Number of Devices: 1

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile S Requirement: Optional

Profile G Requirement: Optional

Profile T Requirement: Optional

7.2.1 Expected Scenarios Under Test:

1. Client connects to Device to Unsubscribe subscriptions.
2. Client is considered as supporting Unsubscribe if the following conditions are met:
 - Client is able to unsubscribe subscriptions using **Unsubscribe** operation.
3. Client is considered as NOT supporting Unsubscribe if the following is TRUE:
 - No valid responses for **Unsubscribe** request OR
 - **Unsubscribe** request does not contains valid **wsa:Action** header.

7.2.2 UNSUBSCRIBE

Test Label: Unsubscribe - Unsubscribe

Test Case ID: UNSUBSCRIBE-1

Feature Under Test: Unsubscribe (Unsubscribe_UnsubscribeAction)

Test Purpose: To verify that the Client is able to use **Unsubscribe** operation to terminate a subscription.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **Unsubscribe** operations present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **Unsubscribe** message with valid **wsa:Action** header to terminete a subscription.

2. Device responses with code HTTP 200 OK and **UnsubscribeResponse** message.

Test Result:**PASS -**

- Client **Unsubscribe** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **Unsubscribe** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **wsnt:Unsubscribe** AND
 - [S2] It contains **wsa:Action** element in header equal to "http://docs.oasis-open.org/wsn/bw-2/SubscriptionManager/UnsubscribeRequest" AND
- Device response on the **Unsubscribe** request fulfills the following requirements:
 - [S3] It has HTTP 200 response code AND
 - [S4] **soapenv:Body** element has child element **wsnt:UnsubscribeResponse**

FAIL -

- The Client failed PASS criteria.

7.3 System Date and Time Configuration Test Cases

7.3.1 Feature Level Requirement:

Validated Feature: System Date and Time Configuration (SystemDateAndTimeConfiguration)

Check Condition based on Device Features: Profile A OR Profile C OR Profile G OR Profile S OR Profile T OR Profile D

Required Number of Devices: 1

Profile A Requirement: Conditional

Profile C Requirement: Optional

Profile G Requirement: Optional

Profile S Requirement: Optional

7.3.2 Expected Scenarios Under Test:

1. Client connects to Device to configure system date and time.

2. Client is considered as supporting System Date and Time Configuration if the following conditions are met:
 - Client is able to retrieve a system date and time using **GetSystemDateAndTime** operation AND
 - Client is able to configure a system date and time using EITHER **SetSystemDateAndTime** operation OR **SetNTP** operation.
3. Client is considered as NOT supporting System Date and Time Configuration if ANY of the following is TRUE:
 - No valid responses for **GetSystemDateAndTime** request OR
 - No valid responses for **SetSystemDateAndTime** request if detected AND
 - Client does not support NTP feature.

7.3.3 GET SYSTEM DATE AND TIME

Test Label: System Date and Time Configuration - Get System Date And Time

Test Case ID: SYSTEMDATEANDTIMECONFIGURATION-1

Feature Under Test: Get System Date And Time
(SystemDateAndTimeConfiguration_GetSystemDateAndTime)

Test Purpose: To verify that Device system date and time is received by Client using the **GetSystemDateAndTime** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetSystemDateAndTime** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetSystemDateAndTime** request message to retrieve system date and time from the Device.
2. Device responds with code HTTP 200 OK and **GetSystemDateAndTimeResponse** message.

Test Result:

PASS -

- Client **GetSystemDateAndTime** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetSystemDateAndTime** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tds:GetSystemDateAndTime** AND
- Device response on the **GetSystemDateAndTime** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **tds:GetSystemDateAndTimeResponse**.

FAIL -

- The Client failed PASS criteria.

7.3.4 SET SYSTEM DATE AND TIME

Test Label: System Date and Time Configuration - Set System Date And Time

Test Case ID: SYSTEMDATEANDTIMECONFIGURATION-2

Feature Under Test: Set System Date And Time
(SystemDateAndTimeConfiguration_SetSystemDateAndTime)

Test Purpose: To verify that Client is able to configure system date and time on Device using the **SetSystemDateAndTime** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetSystemDateAndTime** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetSystemDateAndTime** request message to set Device system date and time.
2. Device responds with code HTTP 200 OK and **SetSystemDateAndTimeResponse** message.

Test Result:**PASS -**

- Client **SetSystemDateAndTime** request messages are valid according to XML Schemas listed in [Namespaces](#) AND

- Client **SetSystemDateAndTime** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tds:SetSystemDateAndTime** AND
 - [S2] If **tds:DateTimeType** element value is equal to "Manual" THEN **tds:SetSystemDateAndTime** contains **tds:UTCDateTime** element AND
- Device response on the **SetSystemDateAndTime** request fulfills the following requirements:
 - [S3] It has HTTP 200 response code AND
 - [S4] **soapenv:Body** element has child element **tds:SetSystemDateAndTimeResponse**.

FAIL -

- The Client failed PASS criteria.

7.4 Hostname Configuration Test Cases

7.4.1 Feature Level Requirement:

Validated Feature: Hostname Configuration (HostnameConfiguration)

Check Condition based on Device Features: None

Required Number of Devices: 1

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile G Requirement: Optional

Profile S Requirement: Optional

7.4.2 Expected Scenarios Under Test:

1. Client connects to Device to configure hostname.
2. Client is considered as supporting Hostname Configuration if the following conditions are met:
 - Client is able to retrieve a hostname information from the Device using **GetHostname** operation AND

- Client is able set a network hostname on the Device using **SetHostname** operation.
3. Client is considered as NOT supporting Hostname Configuration if ANY of the following is TRUE:
- No valid responses for **GetHostname** request OR
 - No valid responses for **SetHostname** request.

7.4.3 GET HOSTNAME

Test Label: Hostname Configuration - Get Hostname

Test Case ID: HOSTNAMECONFIGURATION-1

Feature Under Test: Get Hostname (HostnameConfiguration_GetHostname)

Test Purpose: To verify that hostname settings of the Device are received by Client using the **GetHostname** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetHostname** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetHostname** request message to retrieve hostname from the Device.
2. Device responds with code HTTP 200 OK and **GetHostnameResponse** message.

Test Result:

PASS -

- Client **GetHostname** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetHostname** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tds:GetHostname** AND
- Device response on the **GetHostname** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **tds:GetHostnameResponse**.

FAIL -

- The Client failed PASS criteria.

7.4.4 SET HOSTNAME

Test Label: Hostname Configuration - Set Hostname

Test Case ID: HOSTNAMECONFIGURATION-2

Feature Under Test: Set Hostname (HostnameConfiguration_SetHostname)

Test Purpose: To verify that Client is able to set the Hostname settings on Device using the **SetHostname** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetHostname** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetHostname** request message to set hostname on the Device.
2. Device responds with code HTTP 200 OK and **SetHostnameResponse** message.

Test Result:

PASS -

- Client **SetHostname** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetHostname** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tds:SetHostname** AND
- Device response on the **SetHostname** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **tds:SetHostnameResponse**.

FAIL -

- The Client failed PASS criteria.

7.5 DNS Configuration Test Cases

7.5.1 Feature Level Requirement:

Validated Feature: DNS Configuration (DNSConfiguration)

Check Condition based on Device Features: None

Required Number of Devices: 1

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile G Requirement: Optional

Profile S Requirement: Optional

7.5.2 Expected Scenarios Under Test:

1. Client connects to Device to configure a domain name server.
2. Client is considered as supporting DNS Configuration if the following conditions are met:
 - Client is able to get DNS settings from the Device using **GetDNS** operation AND
 - Client is able set DNS settings on the Device using **SetDNS** operation.
3. Client is considered as NOT supporting DNS Configuration if ANY of the following is TRUE:
 - No valid responses for **GetDNS** request OR
 - No valid responses for **SetDNS** request.

7.5.3 GET DNS

Test Label: DNS Configuration - Get DNS

Test Case ID: DNSCONFIGURATION-1

Feature Under Test: Get DNS (DNSConfiguration_GetDNS)

Test Purpose: To verify that DNS settings of Device are received by Client using the **GetDNS** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetDNS** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetDNS** request message to retrieve DNS settings from the Device.
2. Device responds with code HTTP 200 OK and **GetDNSResponse** message.

Test Result:

PASS -

- Client **GetDNS** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **GetDNS** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tds:GetDNS** AND
- Device response on the **GetDNS** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **tds:GetDNSResponse**.

FAIL -

- The Client failed PASS criteria.

7.5.4 SET DNS

Test Label: DNS Configuration - Set DNS

Test Case ID: DNSCONFIGURATION-2

Feature Under Test: Set DNS (DNSConfiguration_SetDNS)

Test Purpose: To verify that Client is able to set the DNS settings on Device using the **SetDNS** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetDNS** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetDNS** request message to set hostname on the Device.
2. Device responds with code HTTP 200 OK and **SetDNSResponse** message.

Test Result:**PASS -**

- Client **SetDNS** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetDNS** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tds:SetDNS** AND
- Device response on the **SetDNS** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **tds:SetDNSResponse**.

FAIL -

- The Client failed PASS criteria.

7.6 Network Protocols Configuration Test Cases

7.6.1 Feature Level Requirement:

Validated Feature: Network Protocols Configuration (NetworkProtocolsConfiguration)

Check Condition based on Device Features: None

Required Number of Devices: 1

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile G Requirement: Optional

Profile S Requirement: Optional

7.6.2 Expected Scenarios Under Test:

1. Client connects to Device to configure a network protocols.

2. Client is considered as supporting Network Protocols Configuration if the following conditions are met:
 - Client is able to get defined network protocols from the Device using **GetNetworkProtocols** operation AND
 - Client is able configures defined network protocols on the Device using **SetNetworkProtocols** operation.
3. Client is considered as NOT supporting Network Protocols Configuration if ANY of the following is TRUE:
 - No valid responses for **GetNetworkProtocols** request OR
 - No valid responses for **SetNetworkProtocols** request.

7.6.3 GET NETWORK PROTOCOLS

Test Label: Network Protocols Configuration - Get Network Protocols

Test Case ID: NETWORKPROTOCOLSCONFIGURATION-1

Feature Under Test: Get Network Protocols
(NetworkProtocolsConfiguration_GetNetworkProtocols)

Test Purpose: To verify that network protocols of Device are received by Client using the **GetNetworkProtocols** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetNetworkProtocols** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetNetworkProtocols** request message to retrieve network protocols from the Device.
2. Device responds with code HTTP 200 OK and **GetNetworkProtocolsResponse** message.

Test Result:

PASS -

- Client **GetNetworkProtocols** request messages are valid according to XML Schemas listed in [Namespaces](#) AND

- Client **GetNetworkProtocols** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tds:GetNetworkProtocols** AND
- Device response on the **GetNetworkProtocols** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **tds:GetNetworkProtocolsResponse**.

FAIL -

- The Client failed PASS criteria.

7.6.4 SET NETWORK PROTOCOLS

Test Label: Network Protocols Configuration - Set Network Protocols

Test Case ID: NETWORKPROTOCOLSCONFIGURATION-2

Feature **Under** **Test:** Set Network Protocols
(NetworkProtocolsConfiguration_SetNetworkProtocols)

Test Purpose: To verify that Client is able to configure defined network protocols on Device using the **SetNetworkProtocols** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetNetworkProtocols** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetNetworkProtocols** request message to set hostname on the Device.
2. Device responds with code HTTP 200 OK and **SetNetworkProtocolsResponse** message.

Test Result:**PASS -**

- Client **SetNetworkProtocols** request messages are valid according to XML Schemas listed in [Namespaces](#) AND
- Client **SetNetworkProtocols** request in Test Procedure fulfills the following requirements:
 - [S1] **soapenv:Body** element has child element **tds:SetNetworkProtocols** AND

- Device response on the **SetNetworkProtocols** request fulfills the following requirements:
 - [S2] It has HTTP 200 response code AND
 - [S3] **soapenv:Body** element has child element **tds:SetNetworkProtocolsResponse**.

FAIL -

- The Client failed PASS criteria.

8 Supplementary Features and Test Cases

8.1 METADATA STREAMING USING MEDIA2

Test Label: Metadata Streaming Using Media2

Test Case ID: MEDIA2_METADATASTREAMING-1

Feature	Under	Test:	Metadata	Streaming
(Media2_MetadataStreaming_MetadataStreamingUsingMedia2)				

Test Purpose: To verify that the Client is able to retrieve the Metadata Streaming.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with Metadata Streaming using Media2 Service.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetStreamUri** request message for Media2 service for media profile that contains Metadata Configuration. GetStreamUri request is set for RtspUnicast OR RtspMulticast OR RTSP OR RtspOverHttp transport.
2. Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
3. Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
4. Device responds with code RTSP 200 OK and SDP information with Media Type: "application" and with encoding name "vnd.onvif.metadata" or "vnd.onvif.metadata.gzip" or "vnd.onvif.metadata.exi.onvif" or "vnd.onvif.metadata.exi.ext".
5. Client invokes **RTSP SETUP** request without "onvif-replay" Require header and with transport parameter element to to set media session parameters for metadata streaming.
6. Device responds with code RTSP 200 OK.
7. Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
8. Device responds with code RTSP 200 OK.
9. Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.
10. If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:

Note: RTSP requests and RTSP response could be tunneled in HTTP if RtspOverHttp transport is used.

PASS -

- There is Client **RTSP DESCRIBE** request in Test Procedure
- Device response on the **RTSP DESCRIBE** request fulfills the following requirements:
 - [S1] It has RTSP 200 response code AND
 - [S2] SDP packet contains media type "application" (m=application) with sessions attribute "rtmpmap" with encoding name "vnd.onvif.metadata" OR "vnd.onvif.metadata.gzip" OR "vnd.onvif.metadata.exi.onvif" OR "vnd.onvif.metadata.exi.ext" (see ONVIF Streaming Spec) AND
- There is Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
 - [S3] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S4] It invoked after the Client **RTSP DESCRIBE** request AND
 - [S5] RTSP address that was used to send **RTSP SETUP** is correspond to corresponding media Control URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S6] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S7] It has RTSP 200 response code AND
- There is a Device response on the **GetStreamUri** request invoked for Media2 Service in Test Procedure fulfills the following requirements:
 - [S8] It has HTTP 200 response code AND
 - [S9] It received for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S10] It received before the Client **RTSP DESCRIBE** request AND
 - [S11] It contains **tr2:GetStreamUriResponse\tr2:Uri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request AND
- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:

- [S12] It invoked for the same Device as for the Client **RTSP SETUP** request AND
- [S13] It invoked after the Client **RTSP SETUP** request AND
- [S14] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- [S15] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:
 - [S16] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
 - [S17] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S18] It invoked after the Client **RTSP PLAY** request AND
 - [S19] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:
 - [S20] It has RTSP 200 response code.

FAIL -

- The Client failed PASS criteria.

Annex A Test for Appendix A

A.1 Get Video Sources List from GetProfiles responses

Name: HelperGetVideoSourcesListFromGetProfiles

Procedure Purpose: Collect list of video source tokens provided by the device in GetProfiles responses.

Pre-requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetProfiles** operation present.

Input: None

Returns: The complete list of video source tokens detected in all GetProfiles responses (*videoSourcesList*).

Annex Procedure:

- For each **GetProfiles** response detected in the Conversations the Client Test Tool does the following:
 - For each **trt:GetProfilesResponse/trt:Profiles** detected in the Conversations the Client Test Tool does the following:
 - If it contains **VideoSourceConfiguration** element THEN the Client Test Tool adds **tt:VideoSourceConfiguration/tt:SourceToken** value to the *videoSourcesList* if this value does not exists in it.

A.2 Get Video Source Token That was Used for Streaming

Name: HelperGetVideoSourceTokenUsedForStreaming

Procedure Purpose: Get video source token that was used in the media profile requested by the Client in **GetStreamUri** request.

Pre-requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetProfiles** operation or **AddVideoSourceConfiguration** present.

Input: GetStreamUri request

Returns: Video Source token (*videoSourceToken*).

Annex Procedure:

- The Client Test Tool checks that there is Client **AddVideoSourceConfiguration** request or **GetProfiles** response in Test Procedure that fulfills the following requirements:
 - [S1] It is invoked for the same Device as **GetStreamUri** request AND
 - If it is **AddVideoSourceConfiguration** request:
 - [S2] **trt:AddVideoSourceConfiguration/trt:ProfileToken** value is equal to **trt:GetStreamUri/trt:ProfileToken** value AND
 - If it is **GetProfiles** response:
 - [S3] It contains **trt:Profiles** element with **trt:Profiles/@token** value is equal to **trt:GetStreamUri/trt:ProfileToken** value AND
 - [S4] It is the closest one preceding **GetStreamUri** request and it fullfils [S2] or [S3] requirement AND
- The Client Test Tool checks if there is **SetVideoSourceConfiguration** command that fulfills the following requirements:
 - [S5] It invoked for the same Device as **GetStreamUri** request AND
 - If **AddVideoSourceConfiguration** request was found during previous steps:
 - [S6] It invoked after **AddVideoSourceConfiguration** request AND
 - [S7] It is the closest one preceding the **GetStreamUri** request AND
 - [S8] **trt:SetVideoSourceConfiguration/trt:Configuration/@token** value is equal to **trt:AddVideoSourceConfiguration/trt:ConfigurationToken** value AND
 - If **GetProfiles** request was found during previous steps:
 - [S9] It invoked after **GetProfiles** request AND
 - [S10] It is the closest one preceding the **GetStreamUri** request AND
 - [S11] **trt:SetVideoSourceConfiguration/trt:Configuration/@token** value is equal to **tt:VideoSourceConfiguration/@token** value from **trt:GetProfilesResponse/trt:Profiles** with **trt:Profiles/@token** attribute value is equal to **trt:GetStreamUri/trt:ProfileToken** value AND

- IF **SetVideoSourceConfiguration** command was detected during previous steps than **trt:SetVideoSourceConfiguration/trt:Configuration/tt:SourceToken** value will be returned as result of current procedure
- [S12] ELSE IF **GetProfiles** request was found during previous steps then **tt:VideoSourceConfiguration/tt:SourceToken** value from **trt:GetProfilesResponse/trt:Profiles** element with **trt:Profiles/@token** is equal to **trt:GetStreamUri/trt:ProfileToken** value will be returned as result of current procedure
- ELSE IF **AddVideoSourceConfiguration** request was found during previous steps and no **SetVideoSourceConfiguration** was found during previous steps, the Client Test Tool checks the following:
 - There is **GetCompatibleVideoSourceConfigurations** request in Test Procedure that fulfills the following requirements:
 - [S13] It is invoked for the same Device as the **AddVideoSourceConfiguration** request AND
 - [S14] It is the closest one preceding the **AddVideoSourceConfiguration** request AND
 - [S15] **trt:GetCompatibleVideoSourceConfigurations/trt:ProfileToken** value is equal to **trt:GetStreamUri/trt:ProfileToken** value AND
 - Device response on the **GetCompatibleVideoSourceConfigurations** request fulfills the following requirements:
 - [S16] It has HTTP 200 response code AND
 - [S17] **soapenv:Body** element has child element **trt:GetCompatibleVideoSourceConfigurationsResponse** AND
 - [S18] It contains **trt:Configurations/@token** value is equal to **trt:AddVideoSourceConfiguration/trt:ConfigurationToken** value AND
 - [S19] **trt:Configurations/tt:SourceToken** value from **trt:GetCompatibleVideoSourceConfigurationsResponse/trt:Configurations** element with **@token** is equal to **trt:AddVideoSourceConfiguration/trt:ConfigurationToken** value will be returned as result of current procedure

A.3 Find Video Streaming corresponding to GetStreamUri

Name: HelperFindVideoStreamingForGetStreamUri

Procedure Purpose: Find video streaming which corresponding to GetStreamUri pair.

Pre-requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with video streaming present.

Input: GetStreamUri**Returns:** None.**Annex Procedure:**

- There is Client **RTSP DESCRIBE** request in Test Procedure that fulfills the following requirements:
 - [S1] It invoked for the same Device as **GetStreamUri** request AND
 - [S2] It invoked after the Client **GetStreamUri** request AND
 - [S3] RTSP address that was used to send it is equal to **trt:GetStreamUriResponse:trt:MediaUri\|tt:Uri** AND
- Device response on the **RTSP DESCRIBE** request that fulfills the following requirements:
 - [S4] It has RTSP 200 response code AND
 - [S5] SDP packet contains media type "video" (m=video)
- There is Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
 - [S6] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S7] It invoked after the Client **RTSP DESCRIBE** request AND
 - [S8] RTSP address that was used to send **RTSP SETUP** is correspond to corresponding media Control URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S9] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S10] It has RTSP 200 response code AND
- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:
 - [S11] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S13] It invoked after the Client **RTSP SETUP** request AND

- [S14] RTSP address that was used to send it is correspond to video Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- [S15] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:
 - [S16] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
 - [S17] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S18] It invoked after the Client **RTSP PLAY** request AND
 - [S19] RTSP address that was used to send it is correspond to video Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:
 - [S20] It has RTSP 200 response code.

A.4 Get Audio Sources List from GetProfiles responses

Name: HelperGetAudioSourcesListFromGetProfiles

Procedure Purpose: Collect list of audio source tokens provided by the device in GetProfiles responses.

Pre-requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetProfiles** operation present.

Input: None

Returns: The complete list of audio source tokens detected in all GetProfiles responses (*audioSourcesList*).

Annex Procedure:

- For each **GetProfiles** response detected in the Conversations the Client Test Tool does the following:
 - For each **trt:GetProfilesResponse/trt:Profiles** detected in the Conversations the Client Test Tool does the following:
 - If it contains **AudioSourceConfiguration** element THEN the Client Test Tool adds **tt:AudioSourceConfiguration/tt:SourceToken** value to the *audioSourcesList* if this value does not exist in it.

A.5 Get Audio Source Token That was Used for Streaming

Name: HelperGetAudioSourceTokenUsedForStreaming2

Procedure Purpose: Get audio source token that was used in the media profile requested by the Client in **GetStreamUri** request.

Pre-requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetProfiles** operation or **AddAudioSourceConfiguration** present.

Input: GetStreamUri request

Returns: Audio Source token (*audioSourceToken*).

Annex Procedure:

- The Client Test Tool checks that there is Client **AddAudioSourceConfiguration** request or **GetProfiles** response in Test Procedure that fulfills the following requirements:

- [S1] It is invoked for the same Device as **GetStreamUri** request AND

- If it is **AddAudioSourceConfiguration** request:

- [S2] **trt:AddAudioSourceConfiguration/trt:ProfileToken** value is equal to **trt:GetStreamUri/trt:ProfileToken** value AND

If it is **GetProfiles** response:

- [S3] It contains **trt:Profiles** element with **trt:Profiles/@token** value is equal to **trt:GetStreamUri/trt:ProfileToken** value AND
- [S4] It is the closest one preceding **GetStreamUri** request and it fulfills [S2] or [S3] requirement AND

- The Client Test Tool checks if there is **SetAudioSourceConfiguration** command that fulfills the following requirements:
 - [S5] It invoked for the same Device as **GetStreamUri** request AND
 - If **AddAudioSourceConfiguration** request was found during previous steps:
 - [S6] It invoked after **AddAudioSourceConfiguration** request AND
 - [S7] It is the closest one preceding the **GetStreamUri** request AND
 - [S8] **trt:SetAudioSourceConfiguration/trt:ConfigurationToken** value is equal to **trt:AddAudioSourceConfiguration/trt:ConfigurationToken** value AND
 - If **GetProfiles** request was found during previous steps:
 - [S9] It invoked after **GetProfiles** request AND
 - [S10] It is the closest one preceding the **GetStreamUri** request AND
 - [S11] **trt:SetAudioSourceConfiguration/trt:ConfigurationToken** value is equal to **tt:AudioSourceConfiguration** value from **trt:GetProfilesResponse/trt:Profiles** with **@token** attribute value is equal to **trt:GetStreamUri/trt:ProfileToken** value AND
- IF **SetAudioSourceConfiguration** command was detected during previous steps than **trt:SetAudioSourceConfiguration/trt:Configuration/tt:SourceToken** value will be returned as result of current procedure
- [S12] ELSE IF **GetProfiles** request was found during previous steps than **tt:AudioSourceConfiguration/tt:SourceToken** value from **trt:GetProfilesResponse/trt:Profiles** element with **@token** is equal to **trt:GetStreamUri/trt:ProfileToken** value will be returned as result of current procedure
- ELSE IF **AddAudioSourceConfiguration** request was found during previous steps and no **SetAudioSourceConfiguration** was found during previous steps, the Client Test Tool checks the following:
 - There is **GetCompatibleAudioSourceConfigurations** request in Test Procedure that fulfills the following requirements:
 - [S13] It is invoked for the same Device as the **AddAudioSourceConfiguration** request AND
 - [S14] It is the closest one preceding the **AddAudioSourceConfiguration** request AND

- [S15] **trt:GetCompatibleAudioSourceConfigurations/trt:ProfileToken** value is equal to **trt:GetStreamUri/trt:ProfileToken** value AND
- Device response on the **GetCompatibleAudioSourceConfigurations** request fulfills the following requirements:
 - [S16] It has HTTP 200 response code AND
 - [S17] **soapenv:Body** element has child element **trt:GetCompatibleAudioSourceConfigurationsResponse** AND
 - [S18] It contains **trt:Configurations/@token** value is equal to **trt:AddAudioSourceConfiguration/trt:ConfigurationToken** value AND
 - [S19] **trt:Configurations/tt:SourceToken** value from **trt:GetCompatibleAudioSourceConfigurationsResponse/trt:Configurations** element with **@token** is equal to **trt:AddAudioSourceConfiguration/trt:ConfigurationToken** value will be returned as result of current procedure

A.6 Find Audio Streaming corresponding to GetStreamUri

Name: HelperFindAudioStreamingForGetStreamUri

Procedure Purpose: Find audio streaming which corresponding to GetStreamUri pair.

Pre-requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with audio streaming present.

Input: GetStreamUri

Returns: None.

Annex Procedure:

- There is Client **RTSP DESCRIBE** request in Test Procedure that fulfills the following requirements:
 - [S1] It invoked for the same Device as **GetStreamUri** request AND
 - [S2] It invoked after the Client **GetStreamUri** request AND
 - [S3] RTSP address that was used to send it is equal to **trt:GetStreamUriResponse:trt:MediaUri\|tt:Uri** AND
- Device response on the **RTSP DESCRIBE** request that fulfills the following requirements:

- [S4] It has RTSP 200 response code AND
- [S5] SDP packet contains media type "audio" (m=audio)
- There is Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
 - [S6] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
 - [S7] It invoked after the Client **RTSP DESCRIBE** request AND
 - [S8] RTSP address that was used to send **RTSP SETUP** is correspond to corresponding media Control URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S9] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
 - [S10] It has RTSP 200 response code AND
- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:
 - [S11] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S13] It invoked after the Client **RTSP SETUP** request AND
 - [S14] RTSP address that was used to send it is correspond to audio Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
 - [S15] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:
 - [S16] It has RTSP 200 response code AND
- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
 - [S17] It invoked for the same Device as for the Client **RTSP SETUP** request AND
 - [S18] It invoked after the Client **RTSP PLAY** request AND
 - [S19] RTSP address that was used to send it is correspond to audio Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND

- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:
 - [S20] It has RTSP 200 response code.

A.7 Required Number of Devices Summary

Required number of devices and Device feature dependency used in this test specification are listed in the Table.

Table A.1. Required Number of Devices Summary

Feature ID	Feature Name	Required Number of Devices	Check Condition based on Device Features	Check Condition based on Device Features ID
tc.UserToken Profile	Username Token	1 (Note: Username Token feature shall be passed with at least one Device and can be not detected with other devices with supporting of WS-Username Token)	WS-Username Token	WSU
tc.HTTPDigest	HTTP Digest	3	Digest	Digest
tc.Capabilities	Capabilities	3	None	All
tc.MediaStreaming	Media Streaming	3	Real Time Streaming (Media Service) is supported by Device.	RTSS
tc.VideoStreaming	Video Streaming	3	Real Time Streaming (Media Service) is supported by Device.	RTSS
tc.VideoEncoderConfigurations	Video Encoder Configurations	3	Media Service is supported by Device.	MediaService

Feature ID	Feature Name	Required Number of Devices	Check Condition based on Device Features	Check Condition based on Device Features ID
tc.MultipleVideoSources	Multiple Video Sources	3	Real Time Streaming (Media Service) is supported by Device.	RTSS
tc.EventHandling	Event Handling	3	Pull Point Notification OR WS Basic Notification OR Profile S OR Metadata under Media2 service is supported by Device.	no UnsupportedPullPointNotification OR WSBasicNotification OR Profile S OR Media2_Metadata
tc.KeepAliveForPullPointEventHandling	Keep Alive for Pull Point Event Handling	3	Pull Point Notification is supported by Device.	no UnsupportedPullPointNotification
tc.Discovery	Discovery	3	Discovery	All
tc.NVTDiscoveryTypeFilter	Network Video Transmitter Discovery Type Filter	3	Network Video Transmitter Discovery Type is supported by Device.	DiscoveryTypesDnNetworkVideoTransmitter
tc.NetworkConfiguration	Network Configuration	3	Network Configuration	no NetworkConfigNotSupported
tc.System	System	3	None	All
tc.UserHandling	User Handling	3	User Configuration	no UserConfigNotSupported
tc.RelayOutputs	Relay Outputs	1	Relay Outputs (Device Management Service) is supported by Device.	DeviceIORelayOutputs

Feature ID	Feature Name	Required Number of Devices	Check Condition based on Device Features	Check Condition based on Device Features ID
tc.NTP	NTP	1	NTP is supported by Device.	NTP
tc.DynamicDns	Dynamic DNS	1	Dynamic DNS is supported by Device.	DynamicDNS
tc.ZeroConfiguration	Zero Configuration	1	Zero Configuration is supported by Device.	ZeroConfiguration
tc.IPAddressFiltering	IP Address Filtering	1	IP Filter is supported by Device.	IPFilter
tc.MulticastStreaming	Multicast Streaming	1	RTP-Multicast/UDP (Media Service) is supported by Device.	RTPMulticastUDP
tc.MediaProfileConfigurations	Media Profile Configurations	1	Media Service is supported by Device.	MediaService
tc.VideoSourceConfigurations	Video Source Configurations	1	Media Service is supported by Device.	MediaService
tc.AudioStreaming	Audio Streaming	1	Real Time Streaming (Media Service) and Audio (Media Service) are supported by Device.	RTSS AND Audio
tc.MetadataConfigurations	Metadata Configurations	1	Media Service is supported by Device.	MediaService
tc.MultipleAudioSources	Multiple Audio Sources	1	Real Time Streaming (Media Service) and Audio (Media	RTSS AND Audio

Feature ID	Feature Name	Required Number of Devices	Check Condition based on Device Features	Check Condition based on Device Features ID
			Service) are supported by Device.	
tc.PtzListing	PTZ - Listing	1	PTZ Service is supported by Device.	PTZService
tc.PtzConfiguration	PTZ - Configuration	1	PTZ Service and Media Service are supported by Device.	PTZService AND MediaService
tc.PtzPanTiltContinuousPositioning	PTZ Pan Tilt Continuous Positioning	1	PTZ Continuous Pan Tilt movement is supported by Device.	PTZContinuousPanTilt
tc.PtzZoomContinuousPositioning	PTZ Zoom Continuous Positioning	1	PTZ Continuous Zoom movement is supported by Device.	PTZContinuousZoom
tc.PtzPanTiltAbsolutePositioning	PTZ Pan Tilt Absolute Positioning	1	Pan Tilt Absolute Movement and Profile S are supported by Device.	PTZAbsolutePanTilt AND S
tc.PtzZoomAbsolutePositioning	PTZ Zoom Absolute Positioning	1	Zoom Absolute Movement and Profile S are supported by Device.	PTZAbsoluteZoom AND S
tc.PtzPanTiltRelativePositioning	PTZ Pan Tilt Relative Positioning	1	Relative Pan Tilt move and Profile S are supported by Device.	PTZRelativePanTilt AND S
tc.PtzZoomRelativePositioning	PTZ Zoom Relative Positioning	1	Relative Zoom move and Profile S are supported by Device.	PTZRelativeZoom AND S

Feature ID	Feature Name	Required Number of Devices	Check Condition based on Device Features	Check Condition based on Device Features ID
tc.PtzPresets	PTZ Presets	1	PTZ Presets is supported by Device.	PTZPresets
tc.PtzHomePosition	PTZ Home Position	1	PTZ Home Position is supported by Device.	PTZHome
tc.PtzAuxiliaryCommand	PTZ - Auxiliary Command	1	Auxiliary Operations (PTZ Service) and Profile S are supported by Device.	PTZAuxiliary AND S
tc.SetSynchronizationPoint	Set Synchronization Point (Event Service)	1	Pull Point Notification OR WS-Basic Notification is supported by Device.	no UnsupportedPullPointNotification OR WSBasicNotification
tc.SystemDateAndTimeConfiguration	System Date and Time Configuration	1	Profile A OR Profile C OR Profile G OR Profile S OR Profile T OR Profile D	Profile A OR Profile C OR Profile G OR Profile S OR Profile T OR Profile D
tc.HostnameConfiguration	Hostname Configuration	1	None	All
tc.DNSConfiguration	DNS Configuration	1	None	All
tc.NetworkProtocolsConfiguration	Network Protocols Configuration	1	None	All