ONVIF®

Profile G Client Test Specification

Version 22.06

June 2022

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)
21.06	Jan 18, 2021	In the scope of #364 format of the following features were updated to show dependent test cases inside feature: Recording Control Recording Configuration Track Configuration Recording Control – Using a Receiver as Source
20.12	Dec 8, 2020	DEVICEDISCOVERYTYPEFILTER-1 DEVICE 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	Sep 18, 2019	The following was done according to #325: Scope\Supplementary Features and Test Cases sections was added. Supplementary Features and Test Cases sections was added.
19.12	Aug 13, 2019	The following was done according to #325: 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. EVENTHANDLING-4 METADATA STREAMING USING MEDIA was added for Profile S Devices.
19.12	Sep 6, 2019	DEVICEDISCOVERYTYPEFILTER-1 DEVICE DISCOVERY TYPE FILTER was updated according to #323: Unnecessary step with check that ProbeMatch is sent to Client IP address was removed.
19.12	Aug 14, 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 G Client Test Specifications.
19.12	Aug 14, 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 G Client Test Specifications.
19.12	Aug 14, 2019	The following was done according to #341: Get Services section and Get Services Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile G Client Test Specifications.
19.12	Aug 14, 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 G Client Test Specifications.
19.12	Aug 14, 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 G Client Test Specifications.
19.12	Aug 14, 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 G Client Test Specifications.
19.12	Aug 14, 2019	The following was done according to #341:

		Device Discovery Type Filter section and Device Discovery Type Filter Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile G Client Test Specifications.
19.12	Aug 14, 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 G Client Test Specifications.
19.12	Aug 14, 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 G Client Test Specifications.
19.12	Aug 14, 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 G Client Test Specifications.
19.12	Aug 14, 2019	The following was done according to #341: Get Services with Capabilities section and Get Services with Capabilities Test Cases section was moved from ONVIF Core Client Test Specification to ONVIF Profile G Client Test Specifications.
19.12	Aug 14, 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 G Client Test Specifications.
19.12	Aug 14, 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 G Client Test Specifications.
19.12	Aug 14, 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 G Client Test Specifications.
19.12	Aug 14, 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 G Client Test Specifications.
19.12	Aug 14, 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 G Client Test Specifications.
19.12	Aug 14, 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 G 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.
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	Mar 31, 2017	The following test cases were updated according to #168: REPLAYCONTROL-2 MJPEG REPLAY RECORDING REPLAYCONTROL-3 MPEG4 REPLAY RECORDING REPLAYCONTROL-4 H264 REPLAY RECORDING
16.07	Apr 20, 2016	• Test cases about specific event were removed: RECORDINGCONTROL-7, RECORDINGCONTROL-8, RECORDINGCONFIGURATION-5, RECORDINGCONFIGURATION-6, TRACKCONFIGURATION-3, RECEIVER-8, RECEIVER-9.
16.07	Apr 18, 2016	Step description in Test Procedure was updated for the test cases: REPLAYCONTROL-2, REPLAYCONTROL-3, REPLAYCONTROL-4. 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	Mar 18, 2016	Checking of TEARDOWN response was changed in Test Procedure and PASS criteria for the test cases and annexes: REPLAYCONTROL-2, REPLAYCONTROL-3, REPLAYCONTROL-4 Old description of checking of TEARDOWN response in Test Procedure: Device responds with code RTSP 200 OK. New description of checking of TEARDOWN response in Test Procedure: Device response has code RTSP 200 OK if it is detected. Old description of checking of TEARDOWN response in PASS criteria:

		[S32] Device response contains "RTSP/* 200 OK" New description of checking of TEARDOWN response in PASS criteria: [S32] Device response contains "RTSP/* 200 OK" if it is detected
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	REPLAYCONTROL-5 test case: Profile G Requirement was changed from Conditional to Optional.
16.07	Jan 28, 2016	RFC 2326 was added to normative reference. The description about structure and hierarchy was replaced for the test cases: MEDIASEARCH-1, MEDIASEARCH-2, MEDIASEARCH-3, MEDIASEARCH-4, MEDIASEARCH-5, MEDIASEARCH-6, REPLAYCONTROL-1, REPLAYCONTROL-2, REPLAYCONTROL-3, REPLAYCONTROL-4, REPLAYCONTROL-6 Old description: Client %COMMAND NAME% request message is a well-formed SOAP request (refer to onvif.xsd) AND Client %COMMAND NAME% request message has a proper hierarchy (refer to %SERVICE%.wsdl) AND New description: Client %COMMAND NAME% request messages are valid according to XML Schemas listed in Namespaces AND Client %COMMAND NAME% request in Test Procedure fulfills the following requirements: The following steps was removed because the requirements are fullfield by XML Schemas validation: • MEDIASEARCH-1: [S2] "<FindRecordings>" includes tag: "<Scope>" AND [S3] "<FindRecordings>" includes tag: "<KeepAliveTime>" AND [S7] "<GetRecordingSearchResults>" includes tag: "<SearchToken>" AND • MEDIASEARCH-2: [S2] "<FindEvents>" includes tag: "<StartPoint>" AND [S3] "<FindEvents>" includes tag: "<Scope>" AND [S4] "<FindEvents>" includes tag: "<SearchFilter>" AND [S5] "<FindEvents>" includes tag: "<IncludeStartState>" AND [S6] "<FindEvents>" includes tag: "<KeepAliveTime>" AND [S10] "<GetEventSearchResults>" includes tag: "<SearchToken>" AND • MEDIASEARCH-5:

		[S2] "<GetMediaAttributes>" includes tag: "<Time>" AND • MEDIASEARCH-6: [S2] "<FindEvents>" includes tag: "<StartPoint>" AND [S3] "<FindEvents>" includes tag: "<Scope>" AND [S6] "<FindEvents>" includes tag: "<IncludeStartState>" AND [S7] "<FindEvents>" includes tag: "<KeepAliveTime>" AND [S4] "<FindEvents>" includes tag: "<SearchFilter>" AND • REPLAYCONTROL-1: [S3] "<StreamSetup>" includes tag: "<Stream>" with ("RTP-unicast" OR "RTP-multicast") value AND [S2] "<GetReplayUri>" includes tag: "<StreamSetup>" AND [S4] "<StreamSetup>" includes tag: "<Transport>" AND • REPLAYCONTROL-2: [S2] "<GetReplayUri>" includes tag: "<StreamSetup>" AND • REPLAYCONTROL-3: [S2] "<GetReplayUri>" includes tag: "<StreamSetup>" AND • REPLAYCONTROL-4: [S2] "<GetReplayUri>" includes tag: "<StreamSetup>" AND • REPLAYCONTROL-6: [S2] "<SetReplayConfiguration>" includes tag: "<Configuration>" AND [S3] "<Configuration>" includes tag: "<SessionTimeout>" with non-empty value of duration AND
16.01	Nov 25, 2015	General item (Test Overview) was added Minor updates in formatting, typos and terms Updates according review results: • Dynamic Recording Configurations test cases • Dynamic Tracks Configurations test cases • Recording Control test cases • Recording Configuration test cases • Track Configuration test cases • Recording Control – Using a Receiver as Source test cases
16.01	Sep 23, 2015	Dynamic Recording Configurations test cases added Dynamic Tracks Configurations test cases added Recording Control test cases added Namespaces section added Recording Configuration test cases added Track Configuration test cases added Recording Control – Using a Receiver as Source test cases added

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 MEDIASEARCH-6 test case has been updated (removed unnecessary checks of <SearchFilter> tag value).
14.12	Dec 19, 2014	Added Note to Test Result section of MEDIASEARCH-6 Test Case.
14.12	Dec 11, 2014	Fixed typos and inconsistencies.
1.0	Oct 17, 2014	Initial version

Table of Contents

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 G features of a Client application e.g. Recording Search - Media Search. 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 G Client Test Specification defines and regulates the conformance testing procedure for the ONVIF conformant Clients in the scope of Profile G 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 G features according to ONVIF Profile Specifications.

The principal intended purposes are:

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

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

- 3rd parties Client use cases
- 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 and HTTP 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 G Client conformance.

1.2.1 HTTP Digest

HTTP Digest section defines security mechanism for HTTP Digest Authentication.

1.2.2 Capabilities

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

1.2.3 Get Services

Get Services section specifies Client ability to retrieve list of services with using GetServices operation.

1.2.4 Recording Search - Media Search

Recording Search - Media Search section specifies Client ability to perform operations for finding data of interest within a set of recordings or events on Device.

1.2.5 Replay Control

Replay Control section specifies Client ability to control replay of stored video, audio and metadata on Device. This section also specifies Client ability to configure RTSP session timeout.

1.3 Test Cases for Profile Conditional Features

This section defines test cases which are mandatory for Profile G 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 Device Discovery Type Filter

Device Discovery Type Filter Test Cases section defines Client ability to locate services, which are support Device 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 tds:Device 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 Recording Control – Dynamic Recording Configurations

Recording Control – Dynamic Recording Configurations section specifies Client ability to create and delete recordings on Device.

1.3.9 Recording Control – Dynamic Tracks Configurations

Recording Control – Dynamic Tracks Configurations section specifies Client ability to create and delete tracks on Device.

1.3.10 Recording Control

Recording Control section specifies Client ability listing of recordings, managing recording jobs, and managing the state of a recording job on Device. This section also specifies Client ability to retrieve notifications of the change in a recording job's state.

1.3.11 Recording Configuration

Recording Configuration section specifies Client ability get recording configuration, change recording configuration, get recording job configuration, change recording job configuration for Device. This section also specifies Client ability to retrieve notifications of recording configuration change and recording job's configuration change events.

1.3.12 Track Configuration

Track Configuration section specifies Client ability get track configuration, change track configuration for Device. This section also specifies Client ability to retrieve notifications of track configuration change events.

1.3.13 Recording Control – Using a Receiver as Source

Recording Control – Using a Receiver as Source section specifies Client ability listing of receivers, managing receivers, and managing the state and mode of a receivers on Device. This section also specifies Client ability to retrieve notifications of the change in a receivers state and connection failed.

1.4 Test Cases for Profile Optional Features

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

1.4.1 Get Services with Capabilities

Get Services with Capabilities section specifies Client ability to retrieve capabilities of services with using GetServices operation.

1.4.2 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.

1.4.3 Unsubscribe

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

1.4.4 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.

1.4.5 Hostname Configuration

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

1.4.6 DNS Configuration

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

1.4.7 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 G 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/]

- W3C Web Services Addressing 1.0 – Core:

  http://www.w3.org/TR/ws-addr-core/

- ONVIF Profile G Specification:

  https://www.onvif.org/profiles/profile-g/

- 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 G	The Profile G Specification.
Metadata	All streaming data except video and audio, including video analytics results, PTZ position data and other metadata (such as textual data from POS applications).
Recording	A container for a set of audio, video and metadata tracks. A recording can hold one or more tracks. A track is viewed as an infinite timeline that holds data at certain times.
Track	An individual data channel consisting of video, audio, or metadata. This definition is consistent with the definition of track in [RFC 2326].

Video Analytics Algorithms or programs used to analyze video data and to generate data describing object location and behaviour.

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.

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.

Prefix	Namespace URI	Description
wsa	http://www.w3.org/2005/08/addressing	Device addressing namespace as defined by [WS-Addressing].
trv	http://www.onvif.org/ver10/receiver/wsdl	The namespace for the WSDL receiver service
trc	http://www.onvif.org/ver10/recording/wsdl	The namespace for the WSDL recording service
tse	http://www.onvif.org/ver10/search/wsdl	The namespace for the WSDL search service
trp	http://www.onvif.org/ver10/replay/wsdl	The namespace for the WSDL replay 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 G is an ONVIF Client that can configure, request, and control recording of video data over an IP network from an ONVIF Device compliant to the Profile G. The Profile G also includes support for receiving audio and metadata stream if the Client supports those features.

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-reqiusite 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 G, 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 HTTP Digest Test Cases

5.1.1 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

5.1.2 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.

5.1.3 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.

5.2 Capabilities Test Cases

5.2.1 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

5.2.2 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.

5.2.3 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.

5.2.4 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.

5.3 Get Services Test Cases

5.3.1 Feature Level Requirement:

Validated Feature: Get Services (GetServices)

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

Required Number of Devices: 3

Profile A Requirement: Mandatory

Profile D Requirement: Mandatory

Profile C Requirement: Mandatory

Profile G Requirement: Mandatory

Profile T Requirement: Mandatory

Profile M Requirement: Mandatory

5.3.2 Expected Scenarios Under Test:

1. Client connects to Device to retrieve a services using **GetServices** commad.
2. Client is considered as supporting Get Services if the following conditions are met:
   - Client supports Capabilities_GetServicesRequest feature (please see CAPABILITIES-1 GET SERVICES section).
3. Client is considered as NOT supporting Get Services if ANY of the following is TRUE:
   - Client does not support Capabilities_GetServicesRequest feature (please see CAPABILITIES-1 GET SERVICES section).

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.

5.4 Recording Search - Media Search Test Cases

5.4.1 Feature Level Normative Reference:

Validated Feature: Media Search (MediaSearch)

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

Required Number of Devices: 3

Profile G Requirement: Mandatory

5.4.2 Expected Scenarios Under Test:

1. Client connects to Device to start a search session.
2. Client is considered as supporting Recording Search - Media Search if the following conditions are met:
   - Client is able to perform Recording Search using FindRecordings and GetRecordingSearchResults operations AND
   - Client is able to perform Event Search using FindEvents and GetEventSearchResults operations.
3. Client is considered as NOT supporting Recording Search - Media Search if ANY of the following is TRUE:

- No valid responses for FindRecordings OR
- No valid responses for GetRecordingSearchResults OR
- No valid responses for FindEvents OR
- No valid responses for GetEventSearchResults

4. If applicable for Client then any of the following conditions shall be met:
   - Client is able to retrieve Recording Summary using GetRecordingSummary operation OR
   - Client is able to retrieve Recording Information using GetRecordingInformation operation OR
   - Client is able to retrieve Media Attributes using GetMediaAttributes operation OR
   - Client is able to set a search filters using XPath dialect expressions (e.g. for FindEvents operation).

5.4.3 RECORDING SEARCH

Test Label: Media Search - Search for Recordings on Device

Test Case ID: MEDIASEARCH-1

Feature Under Test: Recording Search (MediaSearch_RecordingSearch)

Test Purpose: To verify that the Client is able to perform recordings search session using FindRecordings and GetRecordingSearchResults operations.

Pre-Requisite:

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

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

1. Client invokes FindRecordings request message to starts a search session, looking for recordings that matches the scope.
2. Device responds with code HTTP 200 OK and FindRecordingsResponse message.
3. Client invokes GetRecordingSearchResults request message to receive the results from a recording search session previously initiated by a FindRecordings operation.

4. Device responds with code HTTP 200 OK and GetRecordingSearchResultsResponse message.

Test Result:

PASS -

- Client **FindRecordings** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **FindRecordings** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<FindRecordings>" tag after the "<Body>" tag AND
  - [S4] Device response contains "HTTP/* 200 OK" AND
  - [S5] Device response contains "<FindRecordingsResponse>" tag AND
- Client **GetRecordingSearchResults** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetRecordingSearchResults** request in Test Procedure fulfills the following requirements:
  - [S6] Client request contains "<GetRecordingSearchResults>" tag after the "<Body>" tag AND
  - [S8] Device response contains "HTTP/* 200 OK" AND
  - [S9] Device response contains "<GetRecordingSearchResultsResponse>" tag.

FAIL -

- The Client failed PASS criteria.

5.4.4 EVENT SEARCH

Test Label: Media Search - Search for Events on Device

Test Case ID: MEDIASEARCH-2

Feature Under Test: Event Search (MediaSearch_EventSearch)

Test Purpose: To verify that the Client is able to perform events search session using FindEvents and GetEventSearchResults operations.

Pre-Requisite:

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

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

1. Client invokes FindEvents request message to starts a search session, looking for events that matches the search scope.
2. Device responds with code HTTP 200 OK and FindEventsResponse message.
3. Client invokes GetEventSearchResults request message to receive the results from a recording search session previously initiated by a FindEvents operation.
4. Device responds with code HTTP 200 OK and GetEventSearchResultsResponse message.

Test Result:

PASS -

- Client **FindEvents** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **FindEvents** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<FindEvents>" tag after the "<Body>" tag AND
  - [S7] Device response contains "HTTP/* 200 OK" AND
  - [S8] Device response contains "<FindEventsResponse>" tag AND
- Client **GetEventSearchResults** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetEventSearchResults** request in Test Procedure fulfills the following requirements:
  - [S9] Client request contains "<GetEventSearchResults>" tag after the "<Body>" tag AND
  - [S11] Device response contains "HTTP/* 200 OK" AND
  - [S12] Device response contains "<GetEventSearchResultsResponse>" tag.

FAIL -

- The Client failed PASS criteria.

5.4.5 GET RECORDING SUMMARY

Test Label: Media Search - Get Recording Summary

Test Case ID: MEDIASEARCH-3

Feature Under Test: Recording Summary (MediaSearch_RecordingSummary)

Test Purpose: To verify that Client is able to retrieve Recording Summary using GetRecordingSummary operation.

Pre-Requisite:

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

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

1. Client invokes GetRecordingSummary request message to get a summary description of all recorded data.
2. Device responds with code HTTP 200 OK and GetRecordingSummaryResponse message.

Test Result:

PASS -

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

FAIL -

- The Client failed PASS criteria.

5.4.6 GET RECORDING INFORMATION

Test Label: Media Search - Get Recording Information

Test Case ID: MEDIASEARCH-4

Feature Under Test: Recording Information (MediaSearch_RecordingInformation)

Test Purpose: To verify that Client is able to retrieve Recording Information using GetRecordingInformation operation.

Pre-Requisite:

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

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

1. Client invokes GetRecordingInformation request message to retrieve information about a single Recording specified by a RecordingToken.
2. Device responds with code HTTP 200 OK and GetRecordingInformationResponse message.

Test Result:

PASS -

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

FAIL -

- The Client failed PASS criteria.

5.4.7 GET MEDIA ATTRIBUTES

Test Label: Media Search - Get Media Attributes

Test Case ID: MEDIASEARCH-5

Feature Under Test: Media Attributes (MediaSearch_MediaAttributes)

Test Purpose: To verify that Client is able to retrieve a set of media attributes using GetMediaAttributes operation.

Pre-Requisite:

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

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

1. Client invokes GetMediaAttributes request message to retrieve a set of media attributes for all tracks of the specified recordings at a specified point in time.
2. Device responds with code HTTP 200 OK and GetMediaAttributesResponse message.

Test Result:

PASS -

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

FAIL -

- The Client failed PASS criteria.

5.4.8 FIND EVENTS WITH SEARCH FILTERS

Test Label: Media Search - SearchFilter specified in FindEvents

Test Case ID: MEDIASEARCH-6

Feature Under Test: Event Search Filter (MediaSearch_EventSearchFilter)

Test Purpose: To verify that the Client is able to set a search filters using XPath dialect expressions for FindEvents operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with specified SearchFilter element inside FindEvents request message.

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

1. Client invokes FindEvents request message to start a search session with specified SearchFilter element that contains the topic and message filter needed to define what events to search for.
2. Device responds with code HTTP 200 OK and FindEventsResponse message.

Test Result:

NOTE: If Client FindEvents request message does not contain any value in "<SearchFilter>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **FindEvents** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **FindEvents** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<FindEvents>" tag after the "<Body>" tag AND
  - [S5] "<SearchFilter>" contains any XPath expression AND
  - [S8] Device response contains "HTTP/* 200 OK" AND
  - [S9] Device response contains "<FindEventsResponse>" tag.

FAIL -

- The Client failed PASS criteria.

5.5 Replay Control Test Cases

5.5.1 Feature Level Normative Reference:

Validated Feature: Replay Control (ReplayControl)

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

Required Number of Devices: 3

Profile G Requirement: Mandatory

5.5.2 Expected Scenarios Under Test:

1. Client connects to Device to control replay of stored video, audio and metadata.
2. Client is considered as supporting Replay Control if the following conditions are met:

- Device returns a valid response to GetReplayUri request AND
- Client is able to initiate playback of a recorded stream from Device with either of the following encoding types:
  - MJPEG OR
  - MPEG4 OR
  - H264
- When Device and Client support reverse playback capability for media streaming:
  - Client is able to initiate playback using the negative value of Scale header field in RTSP PLAY command.
- When Device and Client support configurable RTSP session timeout value of the replay service:
  - Client is able to change the value using SetReplayConfiguration operation AND
  - Client is able to retrieve current value using GetReplayConfiguration operation.

3. Client is considered as NOT supporting Replay Control if ANY of the following is TRUE:

- No Valid Device Response to GetReplayUri request OR
- Client is unable to initiate playback of a recorded stream from Device
- When Device and Client support reverse playback capability for media streaming:
  - Client is unable to get valid Device response to RTSP PLAY command with negative value of Scale header field.
- When Device and Client support configurable RTSP session timeout value of the replay service:
  - Client is unable to get valid Device response to SetReplayConfiguration operation OR
  - Client is unable to get valid Device response to GetReplayConfiguration operation.

5.5.3 GET REPLAY URI

Test Label: Replay Control - Get Replay Uri

Test Case ID: REPLAYCONTROL-1

Feature Under Test: Get Replay Uri (ReplayControl_GetReplayUri)

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

Pre-Requisite:

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

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

1. Client invokes GetReplayUri request message with the following Stream Setup: Stream type element with "RTP-unicast" OR "RTP-multicast" value and Transport Protocol element with "UDP" OR "HTTP" OR "RTSP" value and Recording Token element (indicates the media record selected for replay).
2. Device responds with code HTTP 200 OK and GetReplayUriResponse message.

Test Result:

PASS -

- Client **GetReplayUri** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetReplayUri** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<GetReplayUri>" tag after the "<Body>" tag AND
  - [S5] "<Transport>" includes tag: "<Protocol>" with ("UDP" OR "HTTP" OR "RTSP") value AND
  - [S6] "<GetReplayUri>" includes tag: "<RecordingToken>" with non-empty string value of specified record token AND
  - [S7] Device response contains "HTTP/* 200 OK" AND
  - [S8] Device response contains "<GetReplayUriResponse>" tag

FAIL -

- The Client failed PASS criteria.

5.5.4 MJPEG REPLAY RECORDING

Test Label: Replay Control - MJPEG Replay Recording

Test Case ID: REPLAYCONTROL-2

Feature Under Test: MJPEG Replay Recording (ReplayControl_MJPEGReplayRecording)

Test Purpose: To verify that Client is able to replay stored recording from Device by using GetReplayUri operation and RTSP commands to establish and then stop media stream with MJPEG encoding type.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with GetReplayUri operation and RTSP DESCRIBE, RTSP SETUP, RTSP PLAY and RTSP TEARDOWN commands present.

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

1. Client invokes GetReplayUri request message to retrieve URI of stored recording.
2. Device responds with code HTTP 200 OK and GetReplayUriResponse 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 MIME Type: JPEG.
5. Client invokes RTSP SETUP request with transport parameter element to set media session parameters.
6. Device responds with code RTSP 200 OK.
7. Client invokes RTSP PLAY request 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 Device RTSP DESCRIBE response message does not contain Media Type: "video" OR MIME Type: "JPEG" inside SDP information then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **GetReplayUri** request messages are valid according to XML Schemas listed in Namespaces AND

- Client **GetReplayUri** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<GetReplayUri>" tag after the "<Body>" tag AND
  - [S3] "<GetReplayUri>" includes tag: "<RecordingToken>" with non-empty string value of specified record token AND
  - [S4] Device response contains "HTTP/* 200 OK" AND
  - [S5] Device response contains "<GetReplayUriResponse>" tag AND
  - [S6] "<GetReplayUriResponse>" includes tag: "<Uri>" with valid URI address AND
  - [S7] Client request introduces RTSP DESCRIBE command AND
- Client RTSP DESCRIBE request command has a proper hierarchy (see [RFC 2326]) AND
  - [S8] RTSP DESCRIBE includes: URI address obtained from GetReplayUriResponse AND
  - [S9] RTSP DESCRIBE includes: "RTSP/*" version identifier AND
  - [S10] RTSP DESCRIBE includes: "CSeq" identifier AND
  - [S11] Device response contains "RTSP/* 200 OK"
  - [S12] Device response SDP information contains Media Type: "video" and MIME Type: "JPEG" AND
  - [S13] Client request introduces RTSP SETUP command AND
- Client RTSP SETUP request command has a proper hierarchy (see [RFC 2326]) AND
  - [S14] RTSP SETUP includes: URI address AND
  - [S15] RTSP SETUP includes: "RTSP/*" version identifier AND
  - [S16] RTSP SETUP includes: "CSeq" identifier AND
  - [S17] RTSP SETUP includes: "Transport" parameter AND
  - [S18] RTSP SETUP includes: "Require" parameter with "onvif-replay" value AND
  - [S19] Device response contains "RTSP/* 200 OK" AND
  - [S20] Client request introduces RTSP PLAY command AND
- Client RTSP PLAY request command has a proper hierarchy (see [RFC 2326]) AND
  - [S21] RTSP PLAY includes: URI address AND

- [S22] RTSP PLAY includes: "RTSP/*" version identifier AND
- [S23] RTSP PLAY includes: "CSeq" identifier AND
- [S24] RTSP PLAY includes: "Session" parameter AND
- [S25] RTSP PLAY includes: "Require" parameter with "onvif-replay" value AND
- [S26] Device response contains "RTSP/* 200 OK" AND
- [S27] Client request introduces RTSP TEARDOWN command AND

- Client RTSP TEARDOWN request command has a proper hierarchy (see [RFC 2326]) AND
  - [S28] RTSP TEARDOWN includes: URI address AND
  - [S29] RTSP TEARDOWN includes: "RTSP/*" version identifier AND
  - [S30] RTSP TEARDOWN includes: "CSeq" identifier AND
  - [S31] RTSP TEARDOWN includes: "Session" parameter AND
  - [S32] Device response contains "RTSP/* 200 OK" if it is detected.

FAIL -

- The Client failed PASS criteria.

5.5.5 MPEG4 REPLAY RECORDING

Test Label: Replay Control - MPEG4 Replay Recording

Test Case ID: REPLAYCONTROL-3

Feature Under Test: MPEG4 Replay Recording (ReplayControl_MPEG4ReplayRecording)

Test Purpose: To verify that Client is able to replay stored recording from Device by using GetReplayUri operation and RTSP commands to establish and then stop media stream with MPEG4 encoding type.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with GetReplayUri operation and RTSP DESCRIBE, RTSP SETUP, RTSP PLAY and RTSP TEARDOWN commands present.

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

1. Client invokes GetReplayUri request message to retrieve URI of stored recording.
2. Device responds with code HTTP 200 OK and GetReplayUriResponse 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 MIME Type: MPEG4.
5. Client invokes RTSP SETUP request with transport parameter element to set media session parameters.
6. Device responds with code RTSP 200 OK.
7. Client invokes RTSP PLAY request 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 Device RTSP DESCRIBE response message does not contain Media Type: "video" OR MIME Type: "MPEG4" inside SDP information then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **GetReplayUri** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetReplayUri** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<GetReplayUri>" tag after the "<Body>" tag AND
  - [S3] "<GetReplayUri>" includes tag: "<RecordingToken>" with non-empty string value of specified record token AND
  - [S4] Device response contains "HTTP/* 200 OK" AND
  - [S5] Device response contains "<GetReplayUriResponse>" tag AND
  - [S6] "<GetReplayUriResponse>" includes tag: "<Uri>" with valid URI address AND
  - [S7] Client request introduces RTSP DESCRIBE command AND
- Client RTSP DESCRIBE request command has a proper hierarchy (see [RFC 2326]) AND

- [S8] RTSP DESCRIBE includes: URI address obtained from GetReplayUriResponse AND
- [S9] RTSP DESCRIBE includes: "RTSP/*" version identifier AND
- [S10] RTSP DESCRIBE includes: "CSeq" identifier AND
- [S11] Device response contains "RTSP/* 200 OK" AND
- [S12] Device response SDP information contains Media Type: "video" and MIME Type: "MPEG4" AND
- [S13] Client request introduces RTSP SETUP command AND

- Client RTSP SETUP request command has a proper hierarchy (see [RFC 2326]) AND
  - [S14] RTSP SETUP includes: URI address AND
  - [S15] RTSP SETUP includes: "RTSP/*" version identifier AND
  - [S16] RTSP SETUP includes: "CSeq" identifier AND
  - [S17] RTSP SETUP includes: "Transport" parameter AND
  - [S18] RTSP SETUP includes: "Require" parameter with "onvif-replay" value AND
  - [S19] Device response contains "RTSP/* 200 OK" AND
  - [S20] Client request introduces RTSP PLAY command AND
- Client RTSP PLAY request command has a proper hierarchy (see [RFC 2326]) AND
  - [S21] RTSP PLAY includes: URI address AND
  - [S22] RTSP PLAY includes: "RTSP/*" version identifier AND
  - [S23] RTSP PLAY includes: "CSeq" identifier AND
  - [S24] RTSP PLAY includes: "Session" parameter AND
  - [S25] RTSP PLAY includes: "Require" parameter with "onvif-replay" value AND
  - [S26] Device response contains "RTSP/* 200 OK" AND
  - [S27] Client request introduces RTSP TEARDOWN command AND
- Client RTSP TEARDOWN request command has a proper hierarchy (see [RFC 2326]) AND
  - [S28] RTSP TEARDOWN includes: URI address AND

- [S29] RTSP TEARDOWN includes: "RTSP/*" version identifier AND
- [S30] RTSP TEARDOWN includes: "CSeq" identifier AND
- [S31] RTSP TEARDOWN includes: "Session" parameter AND
- [S32] Device response contains "RTSP/* 200 OK" if it is detected.

FAIL -

- The Client failed PASS criteria.

5.5.6 H264 REPLAY RECORDING

Test Label: Replay Control - H264 Replay Recording

Test Case ID: REPLAYCONTROL-4

Feature Under Test: H264 Replay Recording (ReplayControl_H264ReplayRecording)

Test Purpose: To verify that Client is able to replay stored recording from Device by using GetReplayUri operation and RTSP commands to establish and then stop media stream with H264 encoding type.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with GetReplayUri operation and RTSP DESCRIBE, RTSP SETUP, RTSP PLAY and RTSP TEARDOWN commands present.

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

1. Client invokes GetReplayUri request message to retrieve URI of stored recording.
2. Device responds with code HTTP 200 OK and GetReplayUriResponse 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 MIME Type: H264.
5. Client invokes RTSP SETUP request with transport parameter element to set media session parameters.
6. Device responds with code RTSP 200 OK.
7. Client invokes RTSP PLAY request 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 Device RTSP DESCRIBE response message does not contain Media Type: "video" OR MIME Type: "H264" inside SDP information then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **GetReplayUri** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetReplayUri** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<GetReplayUri>" tag after the "<Body>" tag AND
  - [S3] "<GetReplayUri>" includes tag: "<RecordingToken>" with non-empty string value of specified record token AND
  - [S4] Device response contains "HTTP/* 200 OK" AND
  - [S5] Device response contains "<GetReplayUriResponse>" tag AND
  - [S6] "<GetReplayUriResponse>" includes tag: "<Uri>" with valid URI address AND
  - [S7] Client request introduces RTSP DESCRIBE command AND
- Client RTSP DESCRIBE request command has a proper hierarchy (see [RFC 2326]) AND
  - [S8] RTSP DESCRIBE includes: URI address obtained from GetReplayUriResponse AND
  - [S9] RTSP DESCRIBE includes: "RTSP/*" version identifier AND
  - [S10] RTSP DESCRIBE includes: "CSeq" identifier AND
  - [S11] Device response contains "RTSP/* 200 OK" AND
  - [S12] Device response SDP information contains Media Type: "video" and MIME Type: "H264" AND
  - [S13] Client request introduces RTSP SETUP command AND
- Client RTSP SETUP request command has a proper hierarchy (see [RFC 2326]) AND
  - [S14] RTSP SETUP includes: URI address AND

- [S15] RTSP SETUP includes: "RTSP/*" version identifier AND
- [S16] RTSP SETUP includes: "CSeq" identifier AND
- [S17] RTSP SETUP includes: "Transport" parameter AND
- [S18] RTSP SETUP includes: "Require" parameter with "onvif-replay" value AND
- [S19] Device response contains "RTSP/* 200 OK" AND
- [S20] Client request introduces RTSP PLAY command AND

- Client RTSP PLAY request command has a proper hierarchy (see [RFC 2326]) AND
  - [S21] RTSP PLAY includes: URI address AND
  - [S22] RTSP PLAY includes: "RTSP/*" version identifier AND
  - [S23] RTSP PLAY includes: "CSeq" identifier AND
  - [S24] RTSP PLAY includes: "Session" parameter AND
  - [S25] RTSP PLAY includes: "Require" parameter with "onvif-replay" value AND
  - [S26] Device response contains "RTSP/* 200 OK" AND
  - [S27] Client request introduces RTSP TEARDOWN command AND
- Client RTSP TEARDOWN request command has a proper hierarchy (see [RFC 2326]) AND
  - [S28] RTSP TEARDOWN includes: URI address AND
  - [S29] RTSP TEARDOWN includes: "RTSP/*" version identifier AND
  - [S30] RTSP TEARDOWN includes: "CSeq" identifier AND
  - [S31] RTSP TEARDOWN includes: "Session" parameter AND
  - [S32] Device response contains "RTSP/* 200 OK" if it is detected.

FAIL -

- The Client failed PASS criteria.

5.5.7 REVERSE REPLAY

Test Label: Replay Control - Reverse Replay

Test Case ID: REPLAYCONTROL-5

Feature Under Test: Reverse Replay (ReplayControl_ReverseReplay)

Test Purpose: To verify that Client is able to initiate a reverse playback of stored recording from Device by using the negative value of Scale header field in RTSP PLAY command.

Pre-Requisite:

- The Network Trace Capture files contains at least one conversation between Client and Device with RTSP PLAY command with negative value of Scale header field.

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

1. Client invokes RTSP PLAY command with negative value of Scale header field to start reverse playback.
2. Device responds with code RTSP 200 OK.

Test Result:

PASS -

- [S1] Client request introduces RTSP PLAY command AND
- Client RTSP PLAY request command has a proper hierarchy (see [RFC 2326]) AND
  - [S2] RTSP PLAY includes: URI address AND
  - [S3] RTSP PLAY includes: "RTSP/*" version identifier AND
  - [S4] RTSP PLAY includes: "CSeq" identifier AND
  - [S5] RTSP PLAY includes: "Session" parameter AND
  - [S6] RTSP PLAY includes: "Require" parameter with "onvif-replay" value AND
  - [S7] RTSP PLAY includes: "Scale" parameter with any negative value (example: "-1.0", "-2", ...) AND
  - [S8] Device response contains "RTSP/* 200 OK".

FAIL -

- The Client failed PASS criteria.

5.5.8 RTSP SESSION TIMEOUT CONFIGURATION

Test Label: Replay Control - RTSP Session Timeout Configuration

Test Case ID: REPLAYCONTROL-6

Feature Under Test: RTSP Session Timeout Configuration (ReplayControl_RTSPSessionTimeoutConfiguration)

Test Purpose: To verify that Client is able to change the RTSP session timeout configuration of the replay service using SetReplayConfiguration and GetReplayConfiguration operations.

Pre-Requisite:

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

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

1. Client invokes SetReplayConfiguration request with a new duration value of SessionTimeout configuration.
2. Device responds with code HTTP 200 OK and SetReplayConfigurationResponse message.
3. Client invokes GetReplayConfiguration request to verify the current configuration of the replay service.
4. Device responds with code HTTP 200 OK and GetReplayConfigurationResponse message with current configuration listed.

Test Result:

PASS -

- Client **SetReplayConfiguration** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetReplayConfiguration** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<SetReplayConfiguration>" tag after the "<Body>" tag AND
  - [S4] Device response contains "HTTP/* 200 OK" AND
  - [S5] Device response contains "<SetReplayConfigurationResponse>" tag AND
- Client **GetReplayConfiguration** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetReplayConfiguration** request in Test Procedure fulfills the following requirements:
  - [S6] Client request contains "<GetReplayConfiguration>" tag after the "<Body>" tag AND

- [S7] Device response contains "HTTP/* 200 OK" AND
- [S8] Device response contains "<GetReplayConfigurationResponse>" tag.

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 "rtpmap" 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\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 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 subscribtion 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 subscribtion.

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 subscribtion 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 subscribtion.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **CreatePullPointSubscription** operations whithout **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 whithout **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 Device Discovery Type Filter Test Cases

6.4.1 Feature Level Requirement:

Validated Feature: Device Discovery Type Filter (DeviceDiscoveryTypeFilter)

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

Required Number of Devices: 3

Profile S Requirement: None

Profile A Requirement: Mandatory

Profile C Requirement: Conditional

Profile D Requirement: Mandatory

Profile G Requirement: Conditional

Profile T Requirement: Mandatory

Profile M Requirement: Mandatory

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 **tds:Device** or with skipped Types filter.
2. Client is considered as supporting Device Discovery Type if the following conditions are met:
   - **Probe** Client message that fulfills the following requirement is detected:

- Types filter contains tds:Device 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 Device Discovery Type if the following is TRUE:
   - No valid Device **ProbeMatch** message that is correspond to Client **Probe** message.

6.4.3 DEVICE DISCOVERY TYPE FILTER

Test Label: Discovery - Device Discovery Type Filter

Test Case ID: DEVICEDISCOVERYTYPEFILTER-1

Feature Under Test: Device Discovery Type Filter (DeviceDiscoveryTypeFilter_DeviceDiscoveryFilter)

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

Pre-Requisite:

- The Network Trace Capture files contains at least one Client Probe message that does not filter out devices with Device 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 tds:Device.
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 **tds:Device** 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 equeal 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 Recording Control – Dynamic Recording Configurations Test Cases

6.8.1 Feature Level Requirement:

Validated Feature: Dynamic Recording Configuration (DynamicRecordingConfiguration)

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

Required Number of Devices: 1

Profile G Requirement: Conditional

6.8.2 Expected Scenarios Under Test:

1. Client creates new Recording.
2. Client delete a Recording.
3. Client is considered as supporting Dynamic Recording Configurations if the following conditions are met depending on Device features:
   - Client is able to create a recording using the **CreateRecording** operation if Device supports DynamicRecordings feature AND
   - Client is able to delete a recording using the **DeleteRecording** operation if Device supports DynamicRecordings feature.
4. Client is considered as NOT supporting Dynamic Recording Configurations if ANY of the following is TRUE depending on Device features:
   - No Valid Device Response to **CreateRecording** request (except SOAP fault: **soapenv:Receiver/ter:Action/ter:MaxRecordings**) if Device supports DynamicRecordings feature OR

- No Valid Device Response to **DeleteRecording** request (except SOAP fault: **soapenv:Receiver/ter:Action/ter:CannotDelete**) if Device supports DynamicRecordings feature.

6.8.3 CREATE A RECORDING

Test Label: Dynamic Recording Configurations - Create a Recording

Test Case ID: DYNAMICRECORDINGCONFIGURATION-1

Feature Under Test: Create a Recording (DynamicRecordingConfiguration_CreateRecording)

Test Purpose: To verify that Client is able to create a new recording on Device by using the **CreateRecording** operation.

Pre-Requisite:

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

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

1. Client invokes **CreateRecording** request message to create a new recording structure.
2. Device responds with code HTTP 200 OK and **CreateRecordingResponse** message or Device responds with **soapenv:Receiver/ter:Action/ter:MaxRecordings** SOAP fault.

Test Result:

PASS -

- Client **CreateRecording** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **CreateRecording** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **trc:CreateRecording** AND
- Device response on the **CreateRecording** request fulfills the following requirements:
  - [S2] IF it has HTTP 200 response code THEN **soapenv:Body** element has child element **trc:CreateRecordingResponse**
  - [S3] ELSE IF it has other than HTTP 200 response code THEN **soapenv:Body** element has child element **soapenv:Fault** with **soapenv:Receiver/ter:Action/ter:MaxRecordings** fault code.

FAIL -

- The Client failed PASS criteria.

6.8.4 DELETE A RECORDING

Test Label: Dynamic Recording Configurations - Delete a Recording

Test Case ID: DYNAMICRECORDINGCONFIGURATION-2

Feature Under Test: Delete a Recording (DynamicRecordingConfiguration_DeleteRecording)

Test Purpose: To verify that Client is able to delete a recording on Device by using the **DeleteRecording** operation.

Pre-Requisite:

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

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

1. Client invokes **DeleteRecording** request message to delete a recording.
2. Device responds with code HTTP 200 OK and **DeleteRecordingResponse** message or Device responds with **soapenv:Receiver/ter:Action/ter:CannotDelete** SOAP fault.

Test Result:

PASS -

- Client **DeleteRecording** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **DeleteRecording** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **trc:DeleteRecording** AND
  - [S2] **trc:DeleteRecording/trc:RecordingToken** element has non-empty string value of specific recording token AND
- Device response on the **DeleteRecording** request fulfills the following requirements:
  - [S3] IF it has HTTP 200 response code THEN **soapenv:Body** element has child element **trc:DeleteRecordingResponse**

- [S4] ELSE IF it has other than HTTP 200 response code THEN **soapenv:Body** element has child element **soapenv:Fault** with **soapenv:Receiver/ter:Action/ter:CannotDelete** fault code.

FAIL -

- The Client failed PASS criteria.

6.9 Recording Control – Dynamic Track Configurations Test Cases

6.9.1 Feature Level Requirement:

Validated Feature: Dynamic Tracks Configuration (DynamicTracksConfiguration)

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

Required Number of Devices: 1

Profile G Requirement: Conditional

6.9.2 Expected Scenarios Under Test:

1. Client creates new Track.
2. Client delete Track.
3. Client is considered as supporting Dynamic Tracks Configurations if the following conditions are met depending on Device features:
   - Client is able to create a track using the **CreateTrack** operation if Device supports DynamicTracks feature AND
   - Client is able to delete a track using the **DeleteTrack** operation if Device supports DynamicTracks feature.
4. Client is considered as NOT supporting Dynamic Tracks Configurations if ANY of the following is TRUE depending on Device features:
   - No Valid Device Response to **CreateTrack** request (except SOAP fault: **soapenv:Receiver/ter:Action/ter:MaxTracks**) if Device supports DynamicTracks feature OR

- No Valid Device Response to **DeleteTrack** request (except SOAP fault: **soapenv:Receiver/ter:Action/ter:CannotDelete**) if Device supports DynamicTracks feature.

6.9.3 CREATE A TRACK

Test Label: Dynamic Tracks Configurations - Create a Track

Test Case ID: DYNAMICTRACKSCONFIGURATION-1

Feature Under Test: Create a Track (DynamicTracksConfiguration_CreateTrack)

Test Purpose: To verify that Client is able to create a new track within a recording on Device by using the **CreateTrack** operation.

Pre-Requisite:

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

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

1. Client invokes **CreateTrack** request message with specified RecordingToken attribute to create a new track structure.
2. Device responds with code HTTP 200 OK and **CreateTrackResponse** message or Device responds with **soapenv:Receiver/ter:Action/ter:MaxTracks** SOAP fault.

Test Result:

PASS -

- Client **CreateTrack** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **CreateTrack** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **trc:CreateTrack** AND
  - [S2] **trc:CreateTrack/trc:RecordingToken** element has non-empty string value of specific recording token AND
  - [S3] **trc:CreateTrack/trc:TrackConfiguration/tt:TrackType** element value is equal to the strings "Video" OR "Audio" OR "Metadata" AND
- Device response on the **CreateTrack** request fulfills the following requirements:

- [S4] IF it has HTTP 200 response code THEN **soapenv:Body** element has child element **trc:CreateTrackResponse**
- [S5] ELSE IF it has other than HTTP 200 response code THEN **soapenv:Body** element has child element **soapenv:Fault** with **soapenv:Receiver/ter:Action/ter:MaxTracks** fault code.

FAIL -

- The Client failed PASS criteria.

6.9.4 DELETE A TRACK

Test Label: Dynamic Tracks Configurations - Delete a Track

Test Case ID: DYNAMICTRACKSCONFIGURATION-2

Feature Under Test: Delete a Track (DynamicTracksConfiguration_DeleteTrack)

Test Purpose: To verify that Client is able to delete a track within a recoding on Device by using the **DeleteTrack** operation.

Pre-Requisite:

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

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

1. Client invokes **DeleteTrack** request message with specified RecordingToken attribute to delete a track.
2. Device responds with code HTTP 200 OK and **DeleteTrackResponse** message or Device responds with **soapenv:Receiver/ter:Action/ter:CannotDelete** SOAP fault.

Test Result:

PASS -

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

- [S2] **trc:DeleteTrack/trc:RecordingToken** element has non-empty string value of specific recording token AND
- [S3] **trc:DeleteTrack/trc:TrackToken** element has non-empty string value of specific track token AND

- Device response on the **DeleteTrack** request fulfills the following requirements:
  - [S4] IF it has HTTP 200 response code THEN **soapenv:Body** element has child element **trc:DeleteTrackResponse**
  - [S5] ELSE IF it has other than HTTP 200 response code THEN **soapenv:Body** element has child element **soapenv:Fault** with **soapenv:Receiver/ter:Action/ter:CannotDelete** fault code.

FAIL -

- The Client failed PASS criteria.

6.10 Recording Control Test Cases

6.10.1 Feature Level Requirement:

Validated Feature: Recording Control (RecordingControl)

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

Required Number of Devices: 1

Profile G Requirement: Conditional

6.10.2 Expected Scenarios Under Test:

1. Client retrieves a list of recordings using **GetRecordings** operation.
2. Client retrieves a list of recording jobs using **GetRecordingJobs** operation.
3. Client managing recording jobs on a device using **CreateRecordingJob** and **DeleteRecordingJob** operations.
4. Client managing the mode of a recording job on a device using **SetRecordingJobMode** operation.
5. Client may get state of a recording job on a device using **GetRecordingJobState** operation.

6. Client retrieves notifications of the change in a recording job's state using Pull Point event mechanism.
7. When Device and Client support dynamic update of track/recording content capability:
   - Client retrieves notifications of the change in a recording's content using Pull Point event mechanism.
8. Client is considered as supporting Recording Control if the following conditions are met:
   - Client is able to retrieve a list of recordings using the **GetRecordings** operation AND
   - Client is able to retrieve recording jobs using the **GetRecordingJobs** operation AND
   - Client is able to manage recording jobs on a device using the **CreateRecordingJob** and the **DeleteRecordingJob** operations AND
   - Client is able to manage the mode of a recording job on a device using the **SetRecordingJobMode** operation AND
   - Client supports EventHandling_Pullpoint feature (please see EVENTHANDLING-1 PULLPOINT section) AND
   - Client is able to retrieve tns1:RecordingConfig/JobState notification about change of recording job's state AND
   - Client is able to retrieve tns1:RecordingConfig/DeleteTrackData notification about change of recording's content if Device supports RecordingConfigDeleteTrackDataEvent feature.
9. Client is considered as NOT supporting Recording Control if ANY of the following is TRUE:
   - No Valid Device Response to **GetRecordings** request OR
   - No Valid Device Response to **GetRecordingJobs** request OR
   - No Valid Device Response to **CreateRecordingJob** request (except SOAP fault: EITHER **soapenv:Receiver/ter:Action/ter:MaxRecordingJobs** OR **env:Receiver/ter:Action/ter:MaxReceivers**) OR
   - No Valid Device Response to **DeleteRecordingJob** request OR
   - No Valid Device Response to **GetRecordingJobState** request if detected OR
   - No Valid Device Response to **SetRecordingJobMode** request OR
   - Client does not support EventHandling_Pullpoint feature (please see EVENTHANDLING-1 PULLPOINT section) OR

- Client unable to retrieve tns1:RecordingConfig/JobState notification about change of recording job's state OR
- Client unable to retrieve tns1:RecordingConfig/DeleteTrackData notification about change of recording's content if Device supports RecordingConfigDeleteTrackDataEvent feature.

6.10.3 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.

6.10.4 GET RECORDINGS

Test Label: Recording Control - Get Recordings

Test Case ID: RECORDINGCONTROL-1

Feature Under Test: Get Recordings (RecordingControl_GetRecordings)

Test Purpose: To verify that Client is able to receive a list of recordings from Device using the **GetRecordings** operation.

Pre-Requisite:

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

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

1. Client invokes **GetRecordings** request message.

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

Test Result:

PASS -

- Client **GetRecordings** request message is valid according to XML Schemas listed in Namespaces AND

  - [S1] Client request contains "<GetRecordings>" tag after the "<Body>" tag AND

  - [S2] Device response contains "HTTP/* 200 OK" AND

- [S3] Device response contains "<GetRecordingsResponse>" tag

FAIL -

- The Client failed PASS criteria.

6.10.5 GET RECORDING JOBS

Test Label: Recording Control - Get Recording Jobs

Test Case ID: RECORDINGCONTROL-2

Feature Under Test: Get Recording Jobs (RecordingControl_GetRecordingJobs)

Test Purpose: To verify that Client is able to receive a list of recording jobs from Device using the **GetRecordingJobs** operation.

Pre-Requisite:

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

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

1. Client invokes **GetRecordingJobs** request message.
2. Device responds with code HTTP 200 OK and **GetRecordingJobsResponse** message.

Test Result:

PASS -

- Client **GetRecordingJobs** request message is valid according to XML Schemas listed in Namespaces AND
  - [S1] Client request contains "<GetRecordingJobs>" tag after the "<Body>" tag AND
  - [S2] Device response contains "HTTP/* 200 OK" AND
  - [S3] Device response contains "<GetRecordingJobsResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.10.6 GET RECORDING JOB STATE

Test Label: Recording Control - Get Recording Job State

Test Case ID: RECORDINGCONTROL-3

Feature Under Test: Get Recording Job State (RecordingControl_GetRecordingJobState)

Test Purpose: To verify that Client is able to receive a specific recording jobs state from Device using the **GetRecordingJobState** operation.

Pre-Requisite:

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

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

1. Client invokes **GetRecordingJobState** request message.
2. Device responds with code HTTP 200 OK and **GetRecordingJobStateResponse** message.

Test Result:

PASS -

- Client **GetRecordingJobState** request message is valid according to XML Schemas listed in Namespaces AND
  - [S1] Client request contains "<GetRecordingJobState>" tag after the "<Body>" tag AND
  - [S2] "<JobToken>" tag in request has non-empty string value of specific token AND
  - [S3] Device response contains "HTTP/* 200 OK" AND
  - [S4] Device response contains "<GetRecordingJobStateResponse>" tag

FAIL -

- The Client failed PASS criteria.

6.10.7 MODIFY RECORDING JOB MODE

Test Label: Recording Control - Set Recording Job Mode

Test Case ID: RECORDINGCONTROL-4

Feature Under Test: Modify Recording Job Mode (RecordingControl_ModifyRecordingJobMode)

Test Purpose: To verify that Client is able to manage the mode of a recording job on Device using the **SetRecordingJobMode** operation.

Pre-Requisite:

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

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

1. Client invokes **SetRecordingJobMode** request message.
2. Device responds with code HTTP 200 OK and **SetRecordingJobModeResponse** message.

Test Result:

PASS -

- Client **SetRecordingJobMode** request message is valid according to XML Schemas listed in Namespaces AND
  - [S1] Client request contains "<SetRecordingJobMode>" tag after the "<Body>" tag AND
  - [S2] "<JobToken>" tag in request has non-empty string value of specific token AND
  - [S3] "<Mode>" tag in request has value equals EITHER ("Active" OR "Idle") AND
  - [S4] Device response contains "HTTP/* 200 OK" AND
  - [S5] Device response contains "<SetRecordingJobModeResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.10.8 CREATE A RECORDING JOB

Test Label: Recording Control - Create Recording Job

Test Case ID: RECORDINGCONTROL-5

Feature Under Test: Create a Recording Job (RecordingControl_CreateRecordingJob)

Test Purpose: To verify that Client is able to create a new recording job on Device by using the **CreateRecordingJob** operation.

Pre-Requisite:

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

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

1. Client invokes **CreateRecordingJob** request message.
2. Device responds with code HTTP 200 OK and **CreateRecordingJobResponse** message or Device responds with **soapenv:Receiver/ter:Action/ter:MaxRecordingJobs** SOAP fault OR **soapenv:Receiver/ter:Action/ter:MaxReceivers** SOAP fault.

Test Result:

PASS -

- Client **CreateRecordingJob** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **CreateRecordingJob** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **trc:CreateRecordingJob** AND
  - [S2] **trc:CreateRecordingJob/trc:JobConfiguration/tt:RecordingToken** element has non-empty string value of specific recording token AND
  - [S3] **trc:CreateRecordingJob/trc:JobConfiguration/tt:Mode** element has value is equal to EITHER "Active" OR "Idle" AND
  - [S4] **trc:CreateRecordingJob/trc:JobConfiguration/tt:Priority** element has a non-negative value AND
- Device response on the **CreateRecordingJob** request fulfills the following requirements:
  - [S5] IF it has HTTP 200 response code THEN **soapenv:Body** element has child element **trc:CreateRecordingJob**
  - [S6] ELSE IF it has other than HTTP 200 response code THEN **soapenv:Body** element has child element **soapenv:Fault** with **soapenv:Receiver/ter:Action/ter:MaxRecordingJobs** fault code OR **soapenv:Receiver/ter:Action/ter:MaxReceivers** fault code.

FAIL -

- The Client failed PASS criteria.

6.10.9 DELETE A RECORDING JOB

Test Label: Recording Control - Delete Recording Job

Test Case ID: RECORDINGCONTROL-6

Feature Under Test: Delete a Recording Job (RecordingControl_DeleteRecordingJob)

Test Purpose: To verify that Client is able to delete a recording job on Device by using the **DeleteRecordingJob** operation.

Pre-Requisite:

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

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

1. Client invokes **DeleteRecordingJob** request message to delete a recording job.
2. Device responds with code HTTP 200 OK and **DeleteRecordingJobResponse** message.

Test Result:

PASS -

- Client **DeleteRecordingJob** request message is valid according to XML Schemas listed in Namespaces AND
  - [S1] Client request contains "<DeleteRecordingJob>" tag after the "<Body>" tag AND
  - [S2] "<JobToken>" tag in request has non-empty string value of specific recording job token AND
  - [S3] Device response contains "HTTP/* 200 OK" AND
  - [S4] Device response contains "<DeleteRecordingJobResponse>" tag.

FAIL -

- The Client failed PASS criteria.

6.11 Recording Configuration Test Cases

6.11.1 Feature Level Requirement:

Validated Feature: Recording Configuration (RecordingConfiguration)

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

Required Number of Devices: 1

Profile G Requirement: Conditional

6.11.2 Expected Scenarios Under Test:

1. Client retrieves configuration of recording using **GetRecordingConfiguration** operation.
2. Client manages a recording's configuration using **SetRecordingConfiguration** operation.
3. Client retrieves configuration of recording's job using **GetRecordingJobConfiguration** operation.
4. Client manages a recording job's configuration using **SetRecordingJobConfiguration** operation.
5. Client retrieves notifications of the change in a recording's configuration using Pull Point event mechanism, if Device supports recording configuration change event capability.
6. Client retrieves notifications of the change in a recording job's configuration using Pull Point event mechanism, if Device supports recording job's configuration change event capability.
7. Client is considered as supporting Recording Configuration if the following conditions are met:
   - Client is able to retrieve configuration of recording using **GetRecordingConfiguration** operation AND
   - Client is able to manage a recording's configuration using **SetRecordingConfiguration** operation AND
   - Client is able to retrieve configuration of recording's job using **GetRecordingJobConfiguration** operation AND
   - Client is able to manage a recording job's configuration using **SetRecordingJobConfiguration** operation AND
   - Client supports EventHandling_Pullpoint feature (please see EVENTHANDLING-1 PULLPOINT section) AND
   - Client is able to retrieve tns1:RecordingConfig/RecordingConfiguration notification about change in a recording's configuration if Device supports RecordingConfigRecordingConfigurationEvent feature AND
   - Client is able to retrieve tns1:RecordingConfig/RecordingJobConfiguration notification about change in a recording job's configuration if Device supports RecordingConfigRecordingJobConfigurationEvent feature.
8. Client is considered as NOT supporting Recording Configuration if ANY of the following is TRUE:

- No Valid Device Response to **GetRecordingConfiguration** request OR
- No Valid Device Response to **SetRecordingConfiguration** request OR
- No Valid Device Response to **GetRecordingJobConfiguration** request OR
- No Valid Device Response to **SetRecordingJobConfiguration** request OR
- Client does not support EventHandling_Pullpoint feature (please see EVENTHANDLING-1 PULLPOINT section) if Device supports RecordingConfigRecordingConfigurationEvent feature or RecordingConfigRecordingJobConfigurationEvent feature OR
- Client unable to retrieve tns1:RecordingConfig/RecordingConfiguration notification about change of recording's configuration if Device supports RecordingConfigRecordingConfigurationEvent feature OR
- Client unable to retrieve tns1:RecordingConfig/RecordingJobConfiguration notification about change in a recording job's configuration if Device supports RecordingConfigRecordingJobConfigurationEvent feature.

6.11.3 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.

6.11.4 GET RECORDING CONFIGURATION

Test Label: Recording Configuration - Get Recording Configuration

Test Case ID: RECORDINGCONFIGURATION-1

Feature Under Test: Get Recording Configuration (RecordingConfiguration_GetRecordingConfiguration)

Test Purpose: To verify that Client is able to receive configuration of a recording from Device using the **GetRecordingConfiguration** operation.

Pre-Requisite:

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

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

1. Client invokes **GetRecordingConfiguration** request message.
2. Device responds with code HTTP 200 OK and **GetRecordingConfigurationResponse** message.

Test Result:

PASS -

- Client **GetRecordingConfiguration** request message is valid according to XML Schemas listed in Namespaces AND
- Client **GetRecordingConfiguration** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **trc:GetRecordingConfiguration** AND
  - [S2] **trc:RecordingToken** element has non-empty value AND
- Device response on the **GetRecordingConfiguration** request fulfills the following requirements:
  - [S3] It has HTTP 200 response code AND
  - [S4] **soapenv:Body** element has child element **trc:GetRecordingConfigurationResponse**.

FAIL -

- The Client failed PASS criteria.

6.11.5 SET RECORDING CONFIGURATION

Test Label: Recording Configuration - Set Recording Configuration

Test Case ID: RECORDINGCONFIGURATION-2

Feature Under Test: Set Recording Configuration (RecordingConfiguration_SetRecordingConfiguration)

Test Purpose: To verify that Client is able to manages a recording's configuration using the **SetRecordingConfiguration** operation.

Pre-Requisite:

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

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

1. Client invokes **SetRecordingConfiguration** request message.
2. Device responds with code HTTP 200 OK and **SetRecordingConfigurationResponse** message.

Test Result:

PASS -

- Client **SetRecordingConfiguration** request message is valid according to XML Schemas listed in Namespaces AND
- Client **SetRecordingConfiguration** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **trc:SetRecordingConfiguration** AND
  - [S2] **trc:RecordingToken** element has non-empty value AND
- Device response on the **SetRecordingConfiguration** request fulfills the following requirements:
  - [S3] It has HTTP 200 response code AND
  - [S4] **soapenv:Body** element has child element **trc:SetRecordingConfigurationResponse**.

FAIL -

- The Client failed PASS criteria.

6.11.6 GET RECORDING JOB CONFIGURATION

Test Label: Recording Configuration - Get Recording Job Configuration

Test Case ID: RECORDINGCONFIGURATION-3

Feature Under Test: Get Recording Job Configuration (RecordingConfiguration_GetRecordingJobConfiguration)

Test Purpose: To verify that Client is able to receive configuration of a recording job from Device using the **GetRecordingJobConfiguration** operation.

Pre-Requisite:

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

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

1. Client invokes **GetRecordingJobConfiguration** request message.
2. Device responds with code HTTP 200 OK and **GetRecordingJobConfigurationResponse** message.

Test Result:

PASS -

- Client **GetRecordingJobConfiguration** request message is valid according to XML Schemas listed in Namespaces AND
- Client **GetRecordingJobConfiguration** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **trc:GetRecordingJobConfiguration** AND
  - [S2] **trc:JobToken** element has non-empty value AND
- Device response on the **GetRecordingJobConfiguration** request fulfills the following requirements:
  - [S3] It has HTTP 200 response code AND
  - [S4] **soapenv:Body** element has child element **trc:GetRecordingJobConfigurationResponse**.

FAIL -

- The Client failed PASS criteria.

6.11.7 SET RECORDING JOB CONFIGURATION

Test Label: Recording Configuration - Set Recording Job Configuration

Test Case ID: RECORDINGCONFIGURATION-4

Feature Under Test: Set Recording Job Configuration (RecordingConfiguration_SetRecordingJobConfiguration)

Test Purpose: To verify that Client is able to manages a recording job configuration using the **SetRecordingJobConfiguration** operation.

Pre-Requisite:

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

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

1. Client invokes **SetRecordingJobConfiguration** request message.
2. Device responds with code HTTP 200 OK and **SetRecordingJobConfigurationResponse** message.

Test Result:

PASS -

- Client **SetRecordingJobConfiguration** request message is valid according to XML Schemas listed in Namespaces AND
- Client **SetRecordingJobConfiguration** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **trc:SetRecordingJobConfiguration** AND
  - [S2] **trc:JobToken** element has non-empty value AND
  - [S3] **trc:JobConfiguration/tt:RecordingToken** element has non-empty value AND
  - [S4] **trc:JobConfiguration/tt:Mode** element value equals to EITHER "Active" OR "Idle" AND
  - [S5] **trc:JobConfiguration/tt:Priority** element has a non-negative value AND
  - For each **trc:JobConfiguration/tt:Source** element:
    - If does not contain **tt:SourceToken** element then it fulfills the following requirements (else skip the check):
      - [S6] It contains **tt:AutoCreateReceiver** element with value equals to true AND
    - If contains **tt:SourceToken** element then it fulfills the following requirements (else skip the checks):
      - [S7] **tt:SourceToken/tt:Token** element has non-empty string value of specific token AND

- If it contains **tt:SourceToken/@Type** attribute equal to **"http://www.onvif.org/ver10/schema/Profile"** then it fulfills the following requirements (else skip the check):
  - [S8] It does not contain **tt:AutoCreateReceiver** element AND

- Device response on the **SetRecordingJobConfiguration** request fulfills the following requirements:
  - [S9] It has HTTP 200 response code AND
  - [S10] **soapenv:Body** element has child element **trc:SetRecordingJobConfigurationResponse**.

FAIL -

- The Client failed PASS criteria.

6.12 Track Configuration Test Cases

6.12.1 Feature Level Requirement:

Validated Feature: Track Configuration (TrackConfiguration)

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

Required Number of Devices: 1

Profile G Requirement: Conditional

6.12.2 Expected Scenarios Under Test:

1. Client retrieves configuration of track using **GetTrackConfiguration** operation.
2. Client manages a track's configuration using **SetTrackConfiguration** operation.
3. Client retrieves notifications of the change in a track's configuration using Pull Point event mechanism if Device supports track configuration change event capability.
4. Client is considered as supporting Track Configuration if the following conditions are met:
   - Client is able to retrieve configuration of track using **GetTrackConfiguration** operation AND
   - Client is able to manage a track's configuration using **SetTrackConfiguration** operation AND

- Client supports EventHandling_Pullpoint feature (please see EVENTHANDLING-1 PULLPOINT section) AND
- Client is able to retrieve tns1:RecordingConfig/TrackConfiguration notification about change in a track's configuration if Device supports RecordingConfigTrackConfigurationEvent feature.

5. Client is considered as NOT supporting Track Configuration if ANY of the following is TRUE:
   - No Valid Device Response to **GetTrackConfiguration** request OR
   - No Valid Device Response to **SetTrackConfiguration** request OR
   - Client does not support EventHandling_Pullpoint feature (please see EVENTHANDLING-1 PULLPOINT section) if Device supports RecordingConfigTrackConfigurationEvent feature OR
   - Client unable to retrieve tns1:RecordingConfig/TrackConfiguration notification about change of track's configuration if Device supports RecordingConfigTrackConfigurationEvent feature.

6.12.3 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.

6.12.4 GET TRACK CONFIGURATION

Test Label: Track Configuration - Get Track Configuration

Test Case ID: TRACKCONFIGURATION-1

Feature Under Test: Get Track Configuration (TrackConfiguration_GetTrackConfiguration)

Test Purpose: To verify that Client is able to receive configuration of a track from Device using the **GetTrackConfiguration** operation.

Pre-Requisite:

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

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

1. Client invokes **GetTrackConfiguration** request message.
2. Device responds with code HTTP 200 OK and **GetTrackConfigurationResponse** message.

Test Result:

PASS -

- Client **GetTrackConfiguration** request message is valid according to XML Schemas listed in Namespaces AND
- Client **GetTrackConfiguration** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **trc:GetTrackConfiguration** AND
  - [S2] **trc:RecordingToken** element has non-empty value AND
  - [S3] **trc:TrackToken** element has non-empty value AND
- Device response on the **GetTrackConfiguration** request fulfills the following requirements:
  - [S4] It has HTTP 200 response code AND
  - [S5] **soapenv:Body** element has child element **trc:GetTrackConfigurationResponse**.

FAIL -

- The Client failed PASS criteria.

6.12.5 SET TRACK CONFIGURATION

Test Label: Recording Configuration - Set Track Configuration

Test Case ID: TRACKCONFIGURATION-2

Feature Under Test: Set Track Configuration (TrackConfiguration_SetTrackConfiguration)

Test Purpose: To verify that Client is able to manages a track's configuration using the **SetTrackConfiguration** operation.

Pre-Requisite:

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

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

1. Client invokes **SetTrackConfiguration** request message.
2. Device responds with code HTTP 200 OK and **SetTrackConfigurationResponse** message.

Test Result:

PASS -

- Client **SetTrackConfiguration** request message is valid according to XML Schemas listed in Namespaces AND
- Client **SetTrackConfiguration** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **trc:SetTrackConfiguration** AND
  - [S2] **trc:RecordingToken** element has non-empty value AND
  - [S3] **trc:TrackToken** element has non-empty value AND
  - [S4] **trc:TrackConfiguration/tt:TrackType** element value equals EITHER "Video" OR "Audio" OR "Metadata" AND
- Device response on the **SetTrackConfiguration** request fulfills the following requirements:
  - [S5] It has HTTP 200 response code AND
  - [S6] **soapenv:Body** element has child element **trc:SetTrackConfigurationResponse**.

FAIL -

- The Client failed PASS criteria.

6.13 Recording Control – Using a Receiver as Source Test Cases

6.13.1 Feature Level Requirement:

Validated Feature: Receiver (Receiver)

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

Required Number of Devices: 1

Profile G Requirement: Conditional

6.13.2 Expected Scenarios Under Test:

1. Client retrieves a list of receivers from Device using the **GetReceivers** operation.
2. Client retrieves receiver configuration from Device using the **GetReceiver** operation or **GetReceivers** operation.
3. Client creates new receiver using the **CreateReceiver** operation.
4. Client deletes a receiver using the **DeleteReceiver** operation.
5. Client configures a receiver using the **ConfigureReceiver** operation.
6. Client may get state of a recording job on a device using **GetReceiverState** operation.
7. Client retrieves the state of a receiver using subscribes to device messages using **CreatePullPointSubscription** operation to get receiver's state changed notifications.
8. Client retrieves the connection state of a receiver using the subscribes to device messages using **CreatePullPointSubscription** operation to get receiver's state changed notifications OR subscribes to device messages using **CreatePullPointSubscription** operation to get connection failed notifications.
9. Client sets the mode of a receiver using the **SetReceiverMode** operation.
10. Client uses Pull Point event mechanism to retrieve notification events from Device.
11. Client is considered as supporting Using a Receiver as Source if the following conditions are met:
    - Client supports EventHandling_PullPoint feature (please see EVENTHANDLING-1 PULLPOINT section) to use Pull Point event mechanism to retrieve notification events from Device AND
    - Client is able to retrieve list of receivers using **GetReceivers** operation AND
    - Client is able to retrieve receiver configuration using **GetReceiver** operation OR **GetReceivers** operation AND
    - Client is able to create new receiver using **CreateReceiver** operation AND
    - Client is able to delete receiver using **DeleteReceiver** operation AND
    - Client is able to configure receiver using **ConfigureReceiver** operation AND
    - Client is able to retrieve **tns1:Receiver/ChangeState** notification about change in a receiver’s state AND

- Client is able to retrieve **tns1:Receiver/ChangeState** notification OR **tns1:Receiver/ConnectionFailed** notification AND
- Client is able to set receiver mode using **SetReceiverMode** operation.

12. Client is considered as NOT supporting Using a Receiver as Source if ANY of the following is TRUE:
    - Client does not support EventHandling_PullPoint feature (please see EVENTHANDLING-1 PULLPOINT section) OR
    - No Valid Device Response to **GetReceivers** request OR
    - No Valid Device Response to **GetReceiver** request if detected OR
    - No Valid Device Response to **CreateReceiver** request OR
    - No Valid Device Response to **DeleteReceiver** request OR
    - No Valid Device Response to **ConfigureReceiver** request OR
    - No Valid Device Response to **SetReceiverMode** request OR
    - No Valid Device Response to **GetReceiverState** request if detected OR
    - Client unable to retrieve **tns1:Receiver/ChangeState** notification about change in a receiver's state AND
    - Client unable to retrieve **tns1:Receiver/ChangeState** notification AND **tns1:Receiver/ConnectionFailed** notification AND

6.13.3 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.

6.13.4 GET RECEIVERS

Test Label: Recording Control – Using a Receiver as Source - Get Receivers

Test Case ID: RECEIVER-1

Feature Under Test: Get Receivers (Receiver_GetReceivers)

Test Purpose: To verify that list of receivers from Device is received by Client using the **GetReceivers** operation.

Pre-Requisite:

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

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

1. Client invokes **GetReceivers** request message to retrieve list of receivers from the Device.
2. Device responds with code HTTP 200 OK and **GetReceiversResponse** message.

Test Result:

PASS -

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

FAIL -

- The Client failed PASS criteria.

6.13.5 GET RECEIVER

Test Label: Recording Control – Using a Receiver as Source - Get Receiver

Test Case ID: RECEIVER-2

Feature Under Test: Get Receiver (Receiver_GetReceiver)

Test Purpose: To verify that receiver configuration from Device is received by Client using the **GetReceiver** operation.

Pre-Requisite:

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

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

1. Client invokes **GetReceiver** request message to retrieve receiver configuration from the Device.
2. Device responds with code HTTP 200 OK and **GetReceiverResponse** message.

Test Result:

PASS -

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

FAIL -

- The Client failed PASS criteria.

6.13.6 CREATE RECEIVER

Test Label: Recording Control – Using a Receiver as Source - Create Receiver

Test Case ID: RECEIVER-3

Feature Under Test: Create Receiver (Receiver_CreateReceiver)

Test Purpose: To verify that receiver is created on Devicve by Client using the **CreateReceiver** operation.

Pre-Requisite:

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

- Device supports Receiver Service.

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

1. Client invokes **CreateReceiver** request message to create receiver on the Device.
2. Device responds with code HTTP 200 OK and **CreateReceiverResponse** message.

Test Result:

PASS -

- Client **CreateReceiver** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **CreateReceiver** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **trv:CreateReceiver** AND
  - [S2] **trv:Configuration/tt:Mode** element value is not equal **"Unknown"** AND
- Device response on the **CreateReceiver** request fulfills the following requirements:
  - [S3] It has HTTP 200 response code AND
  - [S4] **soapenv:Body** element has child element **trv:CreateReceiverResponse**.

FAIL -

- The Client failed PASS criteria.

6.13.7 DELETE RECEIVER

Test Label: Recording Control – Using a Receiver as Source - Delete Receiver

Test Case ID: RECEIVER-4

Feature Under Test: Delete Receiver (Receiver_DeleteReceiver)

Test Purpose: To verify that receiver is deleted on Devicve by Client using the **DeleteReceiver** operation.

Pre-Requisite:

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

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

1. Client invokes **DeleteReceiver** request message to delete receiver from the Device.
2. Device responds with code HTTP 200 OK and **DeleteReceiverResponse** message.

Test Result:

PASS -

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

FAIL -

- The Client failed PASS criteria.

6.13.8 CONFIGURE RECEIVER

Test Label: Recording Control – Using a Receiver as Source - Configure Receiver

Test Case ID: RECEIVER-5

Feature Under Test: Configure Receiver (Receiver_ConfigureReceiver)

Test Purpose: To verify that receiver is configured on Devicve by Client using the **ConfigureReceiver** operation.

Pre-Requisite:

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

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

1. Client invokes **ConfigureReceiver** request message to configure receiver on the Device.

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

Test Result:

PASS -

- Client **ConfigureReceiver** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **ConfigureReceiver** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **trv:ConfigureReceiver** AND
  - [S2] **trv:Configuration\tt:Mode** element value is not equal **"Unknown"** AND
- Device response on the **ConfigureReceiver** request fulfills the following requirements:
  - [S3] It has HTTP 200 response code AND
  - [S4] **soapenv:Body** element has child element **trv:ConfigureReceiverResponse**.

FAIL -

- The Client failed PASS criteria.

6.13.9 GET RECEIVER STATE

Test Label: Recording Control – Using a Receiver as Source - Get Receiver State

Test Case ID: RECEIVER-6

Feature Under Test: Get Receiver State (Receiver_GetReceiverState)

Test Purpose: To verify that receiver state from Device is received by Client using the **GetReceiverState** operation.

Pre-Requisite:

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

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

1. Client invokes **GetReceiverState** request message to request receiver state from the Device.

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

Test Result:

PASS -

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

FAIL -

- The Client failed PASS criteria.

6.13.10 SET RECEIVER MODE

Test Label: Recording Control – Using a Receiver as Source - Set Receiver Mode

Test Case ID: RECEIVER-7

Feature Under Test: Set Receiver Mode (Receiver_SetReceiverMode)

Test Purpose: To verify that receiver mode is changed by Client on Device using the **SetReceiverMode** operation.

Pre-Requisite:

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

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

1. Client invokes **SetReceiverMode** request message to change receiver mode on the Device.
2. Device responds with code HTTP 200 OK and **SetReceiverModeResponse** message.

Test Result:

PASS -

- Client **SetReceiverMode** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetReceiverMode** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **trv:SetReceiverMode** AND
  - [S2] **trv:Mode** element value is not equal **"Unknown"** AND
- Device response on the **SetReceiverMode** request fulfills the following requirements:
  - [S3] It has HTTP 200 response code AND
  - [S4] **soapenv:Body** element has child element **trv:SetReceiverModeResponse**.

FAIL -

- The Client failed PASS criteria.

7 Test Cases for Profile Optional Features

7.1 Get Services with Capabilities Test Cases

7.1.1 Feature Level Requirement:

Validated Feature: Get Services with Capabilities (GetServicesWithCapabilities)

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

Required Number of Devices: 1

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile D Requirement: Optional

Profile G Requirement: Optional

7.1.2 Expected Scenarios Under Test:

1. Client connects to Device to retrieve a service capabilities.
2. Client is considered as supporting Get Services with Capabilities if the following conditions are met:
   - Client is able to retrieve a services capabilities using **GetServices** operation.
3. Client is considered as NOT supporting Get Services with Capabilities if ANY of the following is TRUE:
   - No valid responses for **GetServices** request.

7.1.3 GET SERVICES

Test Label: Get Services with Capabilities - Get Services

Test Case ID: GETSERVICESWITHCAPABILITIES-1

Feature Under Test: Get Services with Capabilities (GetServicesWithCapabilities_GetServicesWithCapabilitiesRequest)

Test Purpose: To verify that services capabilities provided by Device is received by Client using the **GetServices** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetServices** operation with **tds:IncludeCapability** element equal to true present.
- The Device supports GetServices command.

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

1. Client invokes **GetServices** request message with **tds:IncludeCapability** element equal to true to retrieve redential service capabilities from the Device.
2. Device responds with code HTTP 200 OK and **GetServicesResponse** message.

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] **soapenv:Body** element has child element **tds:GetServices** AND
  - [S2] It contains **tds:IncludeCapability** element equal to true AND
- Device response on the **GetServices** request fulfills the following requirements:
  - [S3] It has HTTP 200 response code AND
  - [S4] **soapenv:Body** element has child element **tds:GetServicesResponse**.

FAIL -

- The Client failed PASS criteria.

7.2 Set Synchronization Point (Event Service) Test Cases

7.2.1 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

7.2.2 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 subscribtions 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.

7.2.3 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 subscribtion.

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 responses 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.

7.3 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

7.3.1 Expected Scenarios Under Test:

1. Client connects to Device to Unsubscribe subscribtions.
2. Client is considered as supporting Unsubscribe if the following conditions are met:
   - Client is able to unsubscribe subscribtions 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.

7.3.2 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 subscribtion.

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.

7.4 System Date and Time Configuration Test Cases

7.4.1 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

7.4.2 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.

7.4.3 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.

7.4.4 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.

7.5 Hostname Configuration Test Cases

7.5.1 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

7.5.2 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.

7.5.3 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.

7.5.4 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.

7.6 DNS Configuration Test Cases

7.6.1 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

7.6.2 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.

7.6.3 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.

7.6.4 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.

7.7 Network Protocols Configuration Test Cases

7.7.1 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

7.7.2 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.

7.7.3 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.

7.7.4 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 "rtpmap" 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 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.HTTPDigest	HTTP Digest	3	Digest	Digest
tc.Capabilities	Capabilities	3	None	All
tc.GetServices	Get Services	3	GetServices is supported by Device.	GetServices
tc.MediaSearch	Recording Search - Media Search	3	Recording Search Service is supported by Device.	RecordingSearchService
tc.ReplayControl	Replay Control	3	Replay Service is supported by Device.	ReplayService
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 UnsupportedPullPointNotificatio
tc.Discovery	Discovery	3	Discovery	All

Feature ID	Feature Name	Required Number of Devices	Check Condition based on Device Features	Check Condition based on Device Features ID
tc.DeviceDiscoveryTypeFilter	Device Discovery Type Filter	3	Device Discovery Type is supported by Device.	DiscoveryTypesTdsDevice
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.DynamicRecordingConfiguration	Recording Control – Dynamic Recording Configurations	1	Dynamic Recordings is supported by Device.	DynamicRecordings
tc.DynamicTracksConfiguration	Recording Control – Dynamic Track Configurations	1	Dynamic Tracks is supported by Device.	DynamicTracks
tc.RecordingControl	Recording Control	1	Recording Control Service is supported by Device.	RecordingControlService
tc.RecordingConfiguration	Recording Configuration	1	Recording Control Service is supported by Device.	RecordingControlService
tc.TrackConfiguration	Track Configuration	1	Recording Control Service is supported by Device.	RecordingControlService
tc.Receiver	Recording Control – Using a Receiver as Source	1	Receiver Service is supported by Device.	ReceiverService
tc.GetServicesWithCapabilities	Get Services with Capabilities	1	GetServices is supported by Device.	GetServices

Feature ID	Feature Name	Required Number of Devices	Check Condition based on Device Features	Check Condition based on Device Features ID
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