

ONVIF[®]

Recording Control Device Test Specification

Version 20.06

June 2020

© 2020 ONVIF, Inc. All rights reserved.

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

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

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

REVISION HISTORY

Vers.	Date	Description
12.12	Dec 20, 2012	First issue of Replay Control Test Specification
13.06	Jun, 2013	<p>The following test cases were updated or added:</p> <p>RECORDING JOB CONFIGURATION - DIFFERENT PRIORITIES (ON MEDIA PROFILE)</p> <p>RECORDING JOB CONFIGURATION - DIFFERENT PRIORITIES (ON RECEIVER)</p> <p>SET RECORDINGS CONFIGURATION (MAXIMUM LENGTH OF RECORDING SOURCE INFORMATION)</p> <p>DYNAMIC RECORDINGS CONFIGURATION (MAXIMUM LENGTH OF RECORDING SOURCE INFORMATION)</p> <p>RECORDING CONTROL – JOB STATE EVENT</p> <p>RECORDING CONTROL – JOB STATE CHANGE EVENT</p> <p>RECORDING CONTROL – RECORDING CONFIGURATION EVENT</p> <p>RECORDING CONTROL – TRACK CONFIGURATION EVENT</p> <p>RECORDING CONTROL – RECORDING JOB CONFIGURATION EVENT</p> <p>RECORDING CONTROL – CREATE RECORDING EVENT</p> <p>RECORDING CONTROL – DELETE RECORDING EVENT</p> <p>RECORDING CONTROL – CREATE TRACK EVENT</p> <p>RECORDING CONTROL – DELETE TRACK EVENT</p> <p>RECORDING CONTROL – CREATE TRACK EVENT (CREATE RECORDING)</p> <p>RECORDING CONTROL – DELETE TRACK EVENT (DELETE RECORDING)</p> <p>RECORDING CONTROL – CREATE TRACK EVENT (CREATE TRACK)</p> <p>RECORDING CONTROL – DELETE TRACK EVENT (DELETE TRACK)</p> <p>RECORDING CONTROL – DELETE TRACK DATA EVENT</p>
13.12	Dec, 2013	<p>The following test cases were updated or added:</p> <p>RECORDING-2-1-1 START RECORDING USING A MEDIA PROFILE was replaced with RECORDING-2-1-10 START RECORDING ON MEDIA PROFILE</p> <p>RECORDING-2-1-2 START RECORDING ON RECEIVER was replaced with RECORDING-2-1-11 START RECORDING ON RECEIVER</p>

RECORDING-2-1-3 STOP RECORDING USING A MEDIA PROFILE - PUT JOB IN IDLE STATE was replaced with RECORDING-2-1-12 STOP RECORDING ON MEDIA PROFILE - PUT JOB IN IDLE STATE

RECORDING-2-1-4 STOP RECORDING ON RECEIVER - PUT JOB IN IDLE STATE was replaced with RECORDING-2-1-13 STOP RECORDING ON RECEIVER - PUT JOB IN IDLE STATE

RECORDING-2-1-5 STOP RECORDING ON RECEIVER - NEVER CONNECTED MODE was replaced with RECORDING-2-1-14 STOP RECORDING ON RECEIVER - NEVER CONNECTED MODE

RECORDING-2-1-6 STOP RECORDING USING A MEDIA PROFILE - DELETE JOB was replaced with RECORDING-2-1-15 STOP RECORDING ON MEDIA PROFILE - DELETE JOB

RECORDING-2-1-7 STOP RECORDING ON RECEIVER - DELETE JOB was replaced with RECORDING-2-1-16 STOP RECORDING ON RECEIVER - DELETE JOB

RECORDING-2-1-8 MODIFY MEDIA ATTRIBUTE WHILE RECORDING - MEDIA PROFILE was replaced with RECORDING-2-1-17 MODIFY MEDIA ATTRIBUTE WHILE RECORDING - MEDIA PROFILE

RECORDING-2-1-9 MODIFY MEDIA ATTRIBUTE WHILE RECORDING - RECEIVER was replaced with RECORDING-2-1-18 MODIFY MEDIA ATTRIBUTE WHILE RECORDING - RECEIVER

RECORDING-3-1-2 DYNAMIC TRACKS CONFIGURATION was replaced with RECORDING-3-1-7 DYNAMIC TRACKS CONFIGURATION

RECORDING-3-1-5 RECORDING JOB CONFIGURATION - DIFFERENT PRIORITIES (ON MEDIA PROFILE) was replaced with RECORDING-3-1-8 RECORDING JOB CONFIGURATION - DIFFERENT PRIORITIES (ON MEDIA PROFILE)

RECORDING-3-1-6 RECORDING JOB CONFIGURATION - DIFFERENT PRIORITIES (ON RECEIVER) was replaced with RECORDING-3-1-9 RECORDING JOB CONFIGURATION - DIFFERENT PRIORITIES (ON RECEIVER)

RECORDING-4-1-6 GET RECORDING JOB CONFIGURATION WITH INVALID TOKEN was replaced with RECORDING-4-1-13 GET RECORDING JOB CONFIGURATION WITH INVALID TOKEN

RECORDING-4-1-8 GET RECORDING JOB STATE WITH INVALID TOKEN was replaced with RECORDING-4-1-14 GET RECORDING JOB STATE WITH INVALID TOKEN

RECORDING-5-1-1 RECORDING CONTROL – JOB STATE EVENT was replaced with RECORDING-5-1-18 RECORDING CONTROL – JOB STATE EVENT

RECORDING-5-1-2 RECORDING CONTROL – JOB STATE CHANGE EVENT was replaced with RECORDING-5-1-19 RECORDING CONTROL – JOB STATE CHANGE EVENT

RECORDING-5-1-5 RECORDING CONTROL – RECORDING JOB CONFIGURATION EVENT was replaced with RECORDING-5-1-20 RECORDING CONTROL – RECORDING JOB CONFIGURATION EVENT

		<p>RECORDING-5-1-7 RECORDING CONTROL – DELETE RECORDING EVENT was replaced with RECORDING-5-1-17 RECORDING CONTROL – DELETE RECORDING EVENT</p> <p>RECORDING-5-1-12 RECORDING CONTROL – CREATE TRACK EVENT (CREATE TRACK) was replaced with RECORDING-5-1-15 RECORDING CONTROL – CREATE TRACK EVENT (CREATE TRACK)</p> <p>RECORDING-5-1-13 RECORDING CONTROL – DELETE TRACK EVENT (DELETE TRACK) was replaced with RECORDING-5-1-16 RECORDING CONTROL – DELETE TRACK EVENT (DELETE TRACK)</p> <p>The following annexes were updated or added:</p> <p>Annex A.6 was replaced with A.12 Selection or Creation of Recording for recording job creation</p> <p>Annex A.9 was replaced with A.13 Auto Creation of Receiver</p> <p>Annex A.14 Selection of Recording for track creation was added</p> <p>Annex A.15 Selection or Creation of Recording for recording job creation on a Media profile was added</p> <p>Annex A.5 was removed.</p>
14.06	Feb, 2014	<p>The following tests were updated with increasing of tests IDs:</p> <p>START RECORDING ON MEDIA PROFILE</p> <p>START RECORDING ON RECEIVER</p> <p>STOP RECORDING ON MEDIA PROFILE - PUT JOB IN IDLE STATE</p> <p>STOP RECORDING ON RECEIVER - PUT JOB IN IDLE STATE</p> <p>STOP RECORDING ON RECEIVER - NEVER CONNECTED MODE</p> <p>STOP RECORDING ON MEDIA PROFILE - DELETE JOB</p> <p>STOP RECORDING ON RECEIVER - DELETE JOB</p> <p>MODIFY MEDIA ATTRIBUTE WHILE RECORDING - MEDIA PROFILE</p> <p>MODIFY MEDIA ATTRIBUTE WHILE RECORDING – RECEIVER</p> <p>DYNAMIC RECORDINGS CONFIGURATION</p> <p>RECORDING JOB CONFIGURATION - DIFFERENT PRIORITIES (ON MEDIA PROFILE)</p> <p>RECORDING JOB CONFIGURATION - DIFFERENT PRIORITIES (ON RECEIVER)</p> <p>GET SERVICES AND GET RECORDING CONTROL SERVICE CAPABILITIES CONSISTENCY</p>
14.12	Dec, 2014	<p>The following annexes were updated:</p> <p>A.16 PullMessages algorithm for check Recording Job State initializing</p>

		<p>The following tests were updated with increasing of tests IDs:</p> <p>START RECORDING ON MEDIA PROFILE</p> <p>STOP RECORDING ON MEDIA PROFILE - PUT JOB IN IDLE STATE</p> <p>STOP RECORDING ON MEDIA PROFILE - DELETE JOB</p> <p>MODIFY MEDIA ATTRIBUTE WHILE RECORDING - MEDIA PROFILE</p>
18.06	Mar, 2018	<p>The following test was removed according #1552:</p> <p>RECORDING-2-1-31 MODIFY MEDIA ATTRIBUTE WHILE RECORDING - MEDIA PROFILE</p>
18.06	June, 2018	Reformatting document using new template
20.06	Dec 18, 2019	<p>The following were updated in the scope of #1435:</p> <p>Reformatting document using new template and format.</p>
20.06	May 13, 2020	<p>Pre-Requisite of the following test cases updated with adding of Pull-Point Notification feature according to #1999:</p> <p>RECORDING-2-1-20 START RECORDING ON RECEIVER</p> <p>RECORDING-2-1-22 STOP RECORDING ON RECEIVER - PUT JOB IN IDLE STATE</p> <p>RECORDING-2-1-23 STOP RECORDING ON RECEIVER - NEVER CONNECTED MODE</p> <p>RECORDING-2-1-25 STOP RECORDING ON RECEIVER - DELETE JOB</p> <p>RECORDING-2-1-27 MODIFY MEDIA ATTRIBUTE WHILE RECORDING - RECEIVER</p> <p>RECORDING-2-1-28 START RECORDING ON MEDIA PROFILE</p> <p>RECORDING-2-1-29 STOP RECORDING ON MEDIA PROFILE - PUT JOB IN IDLE STATE</p> <p>RECORDING-2-1-30 STOP RECORDING ON MEDIA PROFILE - DELETE JOB</p> <p>RECORDING-5-1-3 RECORDING CONTROL – RECORDING CONFIGURATION EVENT</p> <p>RECORDING-5-1-4 RECORDING CONTROL – TRACK CONFIGURATION EVENT</p> <p>RECORDING-5-1-6 RECORDING CONTROL – CREATE RECORDING EVENT</p> <p>RECORDING-5-1-10 RECORDING CONTROL – CREATE TRACK EVENT (CREATE RECORDING)</p> <p>RECORDING-5-1-11 RECORDING CONTROL – DELETE TRACK EVENT (DELETE RECORDING)</p> <p>RECORDING-5-1-14 RECORDING CONTROL – DELETE TRACK DATA EVENT</p>

		<p>RECORDING-5-1-15 RECORDING CONTROL – CREATE TRACK EVENT (CREATE TRACK)</p> <p>RECORDING-5-1-16 RECORDING CONTROL – DELETE TRACK EVENT (DELETE TRACK)</p> <p>RECORDING-5-1-17 RECORDING CONTROL – DELETE RECORDING EVENT</p> <p>RECORDING-5-1-18 RECORDING CONTROL – JOB STATE EVENT</p> <p>RECORDING-5-1-19 RECORDING CONTROL – JOB STATE CHANGE EVENT</p> <p>RECORDING-5-1-20 RECORDING CONTROL – RECORDING JOB CONFIGURATION EVENT</p>
20.06	May 18, 2020	<p>Pre-Requisite of the following test cases updated with adding of Event Service according to #1999:</p> <p>RECORDING-5-1-8 RECORDING CONTROL – CREATE TRACK EVENT</p> <p>RECORDING-5-1-9 RECORDING CONTROL – DELETE TRACK EVENT</p>

Table of Contents

1	Introduction	12
1.1	Scope	12
1.1.1	Capabilities	13
1.1.2	Recording	13
2	Normative references	14
3	Terms and Definitions	16
3.1	Conventions	16
3.2	Definitions	16
3.3	Abbreviations	16
4	Test Overview	18
4.1	Test Setup	18
4.1.1	Network Configuration for DUT	18
4.2	Prerequisites	19
4.3	Test Policy	19
4.3.1	Capabilities	19
4.3.2	Recording	19
4.3.3	Configuration	20
4.3.4	General	20
4.3.5	Authentication Method Selection as a Testing Framework	20
5	Recording Control Test Cases	21
5.1	Capabilities	21
5.1.1	RECORDING CONTROL SERVICE CAPABILITIES	21
5.1.2	GET SERVICES AND GET RECORDING CONTROL SERVICE CAPABILITIES CONSISTENCY	22
5.2	Recording	24
5.2.1	START RECORDING ON RECEIVER	24
5.2.2	STOP RECORDING ON RECEIVER - PUT JOB IN IDLE STATE	26
5.2.3	STOP RECORDING ON RECEIVER - NEVER CONNECTED MODE	29
5.2.4	STOP RECORDING ON RECEIVER - DELETE JOB	32
5.2.5	MODIFY MEDIA ATTRIBUTE WHILE RECORDING - RECEIVER	34

5.2.6	START RECORDING ON MEDIA PROFILE	36
5.2.7	STOP RECORDING ON MEDIA PROFILE - PUT JOB IN IDLE STATE	38
5.2.8	STOP RECORDING ON MEDIA PROFILE - DELETE JOB	41
5.3	Configuration	43
5.3.1	DYNAMIC TRACKS CONFIGURATION	43
5.3.2	DYNAMIC RECORDINGS CONFIGURATION	45
5.3.3	RECORDING JOB CONFIGURATION - DIFFERENT PRIORITIES (ON MEDIA PROFILE)	48
5.3.4	RECORDING JOB CONFIGURATION - DIFFERENT PRIORITIES (ON RECEIVER)	51
5.4	General	53
5.4.1	GET RECORDINGS	53
5.4.2	GET RECORDING CONFIGURATION	54
5.4.3	GET RECORDING CONFIGURATION WITH INVALID TOKEN	55
5.4.4	GET RECORDING JOBS	56
5.4.5	GET RECORDING JOB CONFIGURATION	57
5.4.6	GET RECORDING JOB STATE	59
5.4.7	GET TRACK CONFIGURATION	60
5.4.8	GET TRACK CONFIGURATION WITH INVALID TOKEN	61
5.4.9	SET RECORDINGS CONFIGURATION (MAXIMUM LENGTH OF RECORDING SOURCE INFORMATION)	63
5.4.10	DYNAMIC RECORDINGS CONFIGURATION (MAXIMUM LENGTH OF RECORDING SOURCE INFORMATION)	64
5.4.11	GET RECORDING JOB CONFIGURATION WITH INVALID TOKEN	66
5.4.12	GET RECORDING JOB STATE WITH INVALID TOKEN	68
5.5	Events	69
5.5.1	RECORDING CONTROL – RECORDING CONFIGURATION EVENT	69
5.5.2	RECORDING CONTROL – TRACK CONFIGURATION EVENT	72
5.5.3	RECORDING CONTROL – CREATE RECORDING EVENT	75
5.5.4	RECORDING CONTROL – CREATE TRACK EVENT	78
5.5.5	RECORDING CONTROL – DELETE TRACK EVENT	79

5.5.6	RECORDING CONTROL – CREATE TRACK EVENT (CREATE RECORDING)	80
5.5.7	RECORDING CONTROL – DELETE TRACK EVENT (DELETE RECORDING)	83
5.5.8	RECORDING CONTROL – DELETE TRACK DATA EVENT	85
5.5.9	RECORDING CONTROL – CREATE TRACK EVENT (CREATE TRACK) ..	86
5.5.10	RECORDING CONTROL – DELETE TRACK EVENT (DELETE TRACK)	88
5.5.11	RECORDING CONTROL – DELETE RECORDING EVENT	91
5.5.12	RECORDING CONTROL – JOB STATE EVENT	94
5.5.13	RECORDING CONTROL – JOB STATE CHANGE EVENT	97
5.5.14	RECORDING CONTROL – RECORDING JOB CONFIGURATION EVENT	101
A	Helper Procedures and Additional Notes	106
A.1	Comparison of parameter values for the same recording in GetRecordingsResponse message and in GetRecordingConfigurationResponse message	106
A.2	Comparison of parameter values for the same recording job in GetRecordingJobsResponse message and in GetRecordingJobConfigurationResponse message	106
A.3	Comparison of parameter values for the same recording job in GetRecordingJobsResponse message and in GetRecordingJobStateResponse message	107
A.4	Comparison of parameter values for the same track in GetTrackConfigurationResponse message and in GetRecordingsResponse message ...	107
A.5	PullMessages algorithm for check Recording Job State changing	107
A.6	PullMessages algorithm for check Receiver State changing	108
A.7	Creation of Recording prerequisite	109
A.8	Recording Source Information Parameters Maximum Length	110
A.9	Selection or Creation of Recording for recording job creation	110
A.10	Auto Creation of Receiver	112

- A.11 Selection of Recording for track creation 113
- A.12 Selection or Creation of Recording for recording job creation on a Media profile . 113
- A.13 PullMessages algorithm for check Recording Job State initializing 115

1 Introduction

The goal of the ONVIF test specification set is to make it possible to realize fully interoperable IP physical security implementation from different vendors. The set of ONVIF test specification describes the test cases need to verify the [ONVIF DeviceIO Service Specs] and [ONVIF Conformance] requirements. It also describes the test framework, test setup, pre-requisites, test policies needed for the execution of the described test cases.

This ONVIF Recording Control Test Specification acts as a supplementary document to the [ONVIF Network Interface Specs], illustrating test cases need to be executed and passed. Also, this specification acts as an input document to the development of test tool which will be used to test the ONVIF device implementation conformance towards ONVIF standard. This test tool is referred as ONVIF Client hereafter.

1.1 Scope

This ONVIF Recording Control Test Specification defines and regulates the conformance testing procedure for the ONVIF conformant devices. Conformance testing is meant to be functional black-box testing. The objective of this specification is to provide the test cases to test individual requirements of ONVIF devices according to ONVIF core services which are defined in [ONVIF Network Interface Specs].

The principal intended purposes are:

1. Provide self-assessment tool for implementations.
2. Provide comprehensive test suite coverage for [ONVIF Network Interface Specs].

This specification does not address the following:

1. Product use cases and non-functional (performance and regression) testing.
2. SOAP Implementation Interoperability test i.e. Web Service Interoperability Basic Profile version 2.0 (WS-I BP 2.0).
3. Network protocol implementation Conformance test for HTTP, HTTPS, RTP protocol.
4. Wi-Fi Conformance test

The set of ONVIF Test Specification will not cover the complete set of requirements as defined in [ONVIF Network Interface Specs]; instead it would cover subset of it. The scope of this specification is to derive all the normative requirements of [ONVIF DeviceIO Service Specs] which are related to ONVIF Device IO Service and some of the optional requirements.

This ONVIF Recording Control Test Specification covers ONVIF Recording Control service which is a functional block of [ONVIF Network Interface Specs]. The following sections describe the brief overview and scope of each functional block.

1.1.1 Capabilities

Capabilities test cases are covered for verification to get Recording Control Service capabilities. It means that GetServices and GetServiceCapabilities commands are covered by this test case.

1.1.2 Recording

Recording covers the test cases needed for the verification of recording process as mentioned in [ONVIF Network Interface Specs]. Recording section defines different recording control tests for starting and stopping recording.

2 Normative references

- [ONVIF Conformance] ONVIF Conformance Process Specification:
<https://www.onvif.org/profiles/conformance/>
- [ONVIF Profile Policy] ONVIF Profile Policy:
<https://www.onvif.org/profiles/>
- [ONVIF Network Interface Specs] ONVIF Network Interface Specification documents:
<https://www.onvif.org/profiles/specifications/>
- [ONVIF Core Specs] ONVIF Core Specifications:
<https://www.onvif.org/profiles/specifications/>
- [ONVIF Recording Service Specs] ONVIF Recording Control Specifications:
<https://www.onvif.org/profiles/specifications/>
- [ONVIF Base Test] ONVIF Base Device Test Specification:
<https://www.onvif.org/profiles/conformance/device-test/>
- [ISO/IEC Directives, Part 2] ISO/IEC Directives, Part 2, Annex H:
<http://www.iso.org/directives>
- [ISO 16484-5] ISO 16484-5:2014-09 Annex P:
<https://www.iso.org/obp/ui/#iso:std:63753:en>
- [SOAP 1.2, Part 1] W3C SOAP 1.2, Part 1, Messaging Framework:
<http://www.w3.org/TR/soap12-part1/>
- [XML-Schema, Part 1] W3C XML Schema Part 1: Structures Second Edition:
<http://www.w3.org/TR/xmlschema-1/>
- [XML-Schema, Part 2] W3C XML Schema Part 2: Datatypes Second Edition:
<http://www.w3.org/TR/xmlschema-2/>
- [WS-Security] "Web Services Security: SOAP Message Security 1.1 (WS-Security 2004)", OASIS Standard, February 2006.:

<http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>

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 defines terms that are specific to the ONVIF Receiver Service and tests. For the list of applicable general terms and definitions, please see [ONVIF Base Test].

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.
Recording Event	An event associated with a Recording, represented by a notification message in the APIs
Recording Job	A job performs the transfer of data from a data source to a particular recording using a particular configuration
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 behavior.

3.3 Abbreviations

This section describes abbreviations used in this document.

DUT	Device Under Test
HTTP	Hyper Text Transport Protocol
RTCP	RTP Control Protocol
RTSP	Real Time Streaming Protocol
RTP	Real-time Transport Protocol
SDP	Session Description Protocol
TCP	Transport Control Protocol
UTC	Coordinated Universal Time
UDP	User Datagram Protocol

URI	Uniform Resource Identifier
WSDL	Web Services Description Language
WS-I BP 2.0	Web Services Interoperability Basic Profile version 2.0
XML	eXtensible Markup Language

4 Test Overview

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

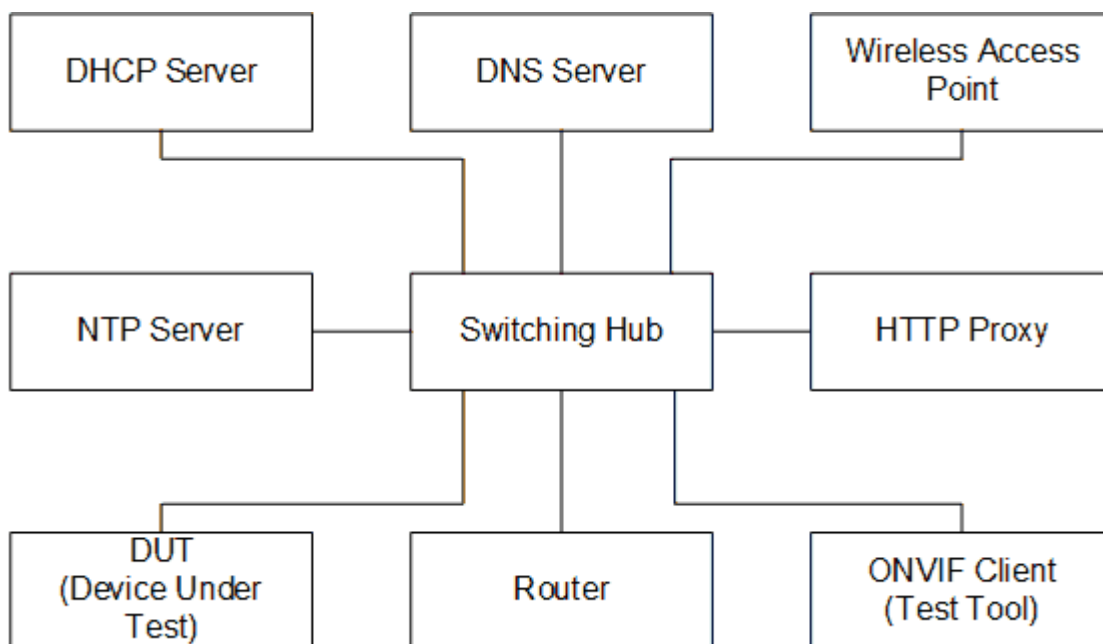
4.1 Test Setup

4.1.1 Network Configuration for DUT

The generic test configuration for the execution of test cases defined in this document is as shown below (Figure 4.1).

Based on the individual test case requirements, some of the entities in the below setup may not be needed for the execution of those corresponding test cases.

Figure 4.1. Test Configuration for DUT



DUT: ONVIF device to be tested. Hereafter, this is referred to as DUT (Device Under Test).

ONVIF Client (Test Tool): Tests are executed by this system and it controls the behavior of the DUT. It handles both expected and unexpected behavior.

HTTP Proxy: provides facilitation in case of RTP and RTSP tunneling over HTTP.

Wireless Access Point: provides wireless connectivity to the devices that support wireless connection.

DNS Server: provides DNS related information to the connected devices.

DHCP Server: provides IPv4 Address to the connected devices.

NTP Server: provides time synchronization between ONVIF Client and DUT.

Switching Hub: provides network connectivity among all the test equipments in the test environment. All devices should be connected to the Switching Hub. When running multiple test instances in parallel on the same network, the Switching Hub should be configured to use filtering in order to avoid multicast traffic being flooded to all ports, because this may affect test stability.

Router: provides router advertisements for IPv6 configuration.

4.2 Prerequisites

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

1. The DUT shall be configured with an IPv4 address.
2. The DUT shall be IP reachable [in the test configuration].
3. The DUT shall be able to be discovered by the Test Tool.
4. The DUT shall be configured with the time, i.e. manual configuration of UTC time and if NTP is supported by the DUT then NTP time shall be synchronized with NTP Server.
5. The DUT time and Test tool time shall be synchronized with each other either manually or by a common NTP server.

4.3 Test Policy

This section describes the test policies specific to the test case execution of each functional block.

The DUT shall adhere to the test policies defined in this section.

4.3.1 Capabilities

The device under test shall demonstrate recording control service capability in GetServices and GetServiceCapabilities responses. A DUT that does not display recording control service capability constitutes failure of test procedure.

Please refer to [Section 5.1](#) for Capabilities Test Cases.

4.3.2 Recording

The DUT shall give the Recording Control Service entry point by GetServices command.

Please refer to [Section 5.2](#) for Recording Test Cases.

4.3.3 Configuration

The DUT shall give the Recording Control Service entry point by GetServices command.

Please refer to [Section 5.3](#) for Configuration Test Cases.

4.3.4 General

The DUT shall give the Recording Control Service entry point by GetServices command.

Please refer to [Section 5.4](#) for General Test Cases.

4.3.5 Authentication Method Selection as a Testing Framework

According to later version of [ONVIF Network Interface Specs], it requires ONVIF client to support both HTTP digest and WS-UsernameToken functionality as authentication functionality. Therefore, ONVIF Client (ONVIF Device Test Tool in this context) as a testing framework shall properly select authentication method between the two based on the response from DUT toward specific request. The following is the deterministic procedure on which authentication method is to be selected.

Procedure:

1. ONVIF Client invokes a specific command which is under testing without any user credentials (no WS-UsernameToken, no HTTP digest authentication header).
2. If DUT returns correct response, then ONVIF Client determines that DUT does not require any user authentication toward the command according to the configured security policy.
3. If DUT returns HTTP 401 Unauthorized error along with WWW-Authentication: Digest header, then ONVIF Client determines that DUT supports HTTP digest authentication. ONVIF Client shall provide with the proper level of user credential to continue the test procedure.
4. If the DUT returns SOAP fault (Sender/NotAuthorized) message, then ONVIF Client determines that WS-UsernameToken is supported by DUT. ONVIF Client shall provide with the proper level of user credential to continue the test procedure.

5 Recording Control Test Cases

5.1 Capabilities

5.1.1 RECORDING CONTROL SERVICE CAPABILITIES

Test Case ID: RECORDING-1-1-1

Specification Coverage: Capability exchange (ONVIF Core Specification), GetServiceCapabilities (ONVIF Recording Control Service Specification)

Feature Under Test: GetServiceCapabilities (for Recording Control Service)

WSDL Reference: recording.wsdl

Test Purpose: To verify Recording Control Service Capabilities of the DUT.

Pre-Requisite: ONVIF Client gets the Recording Control Service entry point by GetServices/GetCapabilities command.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetServiceCapabilitiesRequest** message to retrieve recording control service capabilities of the DUT.
4. Verify the **GetServiceCapabilitiesResponse** from the DUT.

Test Result:

PASS –

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetServiceCapabilitiesResponse**.
- The DUT sent Capabilities.MaxRate zero or less.

- The DUT sent Capabilities.MaxTotalRate zero or less.
- The DUT sent Capabilities.MaxRecordings less than 1.
- The DUT sent an empty Capabilities.Encoding list.

5.1.2 GET SERVICES AND GET RECORDING CONTROL SERVICE CAPABILITIES CONSISTENCY

Test Case ID: RECORDING-1-1-3

Specification Coverage: Capability exchange (ONVIF Core Specification), GetServiceCapabilities (ONVIF Recording Control Service Specification)

Feature Under Test: GetServices, GetServiceCapabilities (for Recording Control Service)

WSDL Reference: devicemgmt.wsdl, recording.wsdl

Test Purpose: To verify Get Services and Recording Control Service Capabilities consistency.

Pre-Requisite: None.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetServicesRequest** message (IncludeCapability = true) to retrieve all services of the DUT with service capabilities.
4. Verify the **GetServicesResponse** message from the DUT.
5. ONVIF Client will invoke **GetServiceCapabilitiesRequest** message to retrieve Recording Control service capabilities of the DUT.
6. Verify the **GetServiceCapabilitiesResponse** message from the DUT.

Test Result:

PASS –

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetServicesResponse** message.
- The DUT did not send a valid **GetServiceCapabilitiesResponse** message.
- The DUT sent different Capabilities in **GetServicesResponse** message and in **GetServiceCapabilitiesResponse** message.

Note: Service will be defined as Recording Control service if it has Namespace element that is equal to "http://www.onvif.org/ver10/recording/wsdl".

Note: Capabilities in **GetServicesResponse** message and in **GetServiceCapabilitiesResponse** message will be assumed as different in the following cases:

- DynamicRecordings attribute is skipped only for **GetServicesResponse** message or only for **GetServiceCapabilitiesResponse** message.
- DynamicRecordings attribute values are different.
- DynamicTracks attribute is skipped only for **GetServicesResponse** message or only for **GetServiceCapabilitiesResponse** message.
- DynamicTracks attribute values are different.
- Encoding attribute is skipped only for **GetServicesResponse** message or only for **GetServiceCapabilitiesResponse** message.
- Encoding attribute values are different.
- MaxRate attribute is skipped only for **GetServicesResponse** message or only for **GetServiceCapabilitiesResponse** message.
- MaxRate attribute values are different.
- MaxTotalRate attribute is skipped only for **GetServicesResponse** message or only for **GetServiceCapabilitiesResponse** message.
- MaxTotalRate attribute values are different.
- MaxRecordings attribute is skipped only for **GetServicesResponse** message or only for **GetServiceCapabilitiesResponse** message.
- MaxRecordings attribute values are different.
- MetadataRecording attribute is skipped only for **GetServicesResponse** message or only for **GetServiceCapabilitiesResponse** message.
- MetadataRecording attribute values are different.

- Options attribute is skipped only for **GetServicesResponse** message or only for **GetServiceCapabilitiesResponse** message.
- Options attribute values are different.

5.2 Recording

5.2.1 START RECORDING ON RECEIVER

Test Case ID: RECORDING-2-1-20

Specification Coverage: Creation of Recording Job with Active mode on receiver. Event generation when the state field of the RecordingJobStateInformation structure is changing (ONVIF Recording Control Service Specification)

Feature Under Test: CreateRecordingJob

WSDL Reference: recording.wsdl, receiver.wsdl, event.wsdl

Test Purpose: To verify Start Recording on Receiver.

Pre-Requisite: ONVIF Client gets the entry points for Recording Control Service and Receiver Service by GetServices command. All recording jobs were stopped. Options are supported by DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.9](#) with RequiredSpareJobs = 0 to create or select Recording to make sure that 1 recording job can be created.
4. Execute [Annex A.10](#) for auto creation of receiver by create recording job with Idle mode.
5. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/JobState Topic as Filter and an InitialTerminationTime = TerminationTime1 to check Recording Job State changing.
6. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
7. ONVIF Client will invoke **ConfigureReceiverRequest** message (ReceiverToken = ReceiverToken1, Configuration.Mode = "AlwaysConnect", Configuration.MediaUri

- as `stream_uri` of RTSP Simulator, `Configuration.StreamSetup.Stream = "RTP-Unicast"`, `StreamSetup.Transport.Tunnel.Protocol = "UDP"`, no `StreamSetup.Transport.Tunnel.Tunnel`) to configure the receiver to receive media from RTSP Simulator.
8. Verify **ConfigureReceiverResponse** message from the DUT.
 9. ONVIF Client will invoke **GetReceiverRequest** message with `ReceiverToken = ReceiverToken1`.
 10. Verify **GetReceiverResponse** message from the DUT. Check that **GetReceiverResponse** message contains the same parameters values as the ones changed in **ConfigureReceiverRequest** message.
 11. ONVIF Client will invoke **SetRecordingJobModeRequest** message (`JobToken = "JobToken1"`, `Mode = "Active"`) to start recording.
 12. Verify **SetRecordingJobModeResponse** message from the DUT. Mark time of **CreateRecordingJobResponse** as T1.
 13. Execute [Annex A.5](#) for receiving event with "Active" or "PartiallyActive" Recording job state.
 14. ONVIF Client will invoke **GetRecordingJobStateRequest** message with `JobToken = "JobToken1"`.
 15. Verify the **GetRecordingJobStateResponse** message from the DUT.
 16. Check that `State.State` is equal to "Active" or "PartiallyActive".
 17. Check that `State.Sources.State` is equal to "Active".
 18. Check that `State.State` value in the **GetRecordingJobStateResponse** message equals to `State.State` value in the notification message.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **CreatePullPointSubscriptionRequest** message
- The DUT did not send a valid **ConfigureReceiverResponse** message.
- The DUT did not send a valid **GetReceiverResponse** message.

- The DUT did not send a valid **SetRecordingJobModeResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message.
- The DUT sent **GetReceiverResponse** message with parameters values differ from sent in **ConfigureReceiverRequest** message.
- The DUT sent **GetRecordingJobStateResponse** message with State.State parameter value not equal to "Active" or "PartiallyActive".
- The DUT sent **GetRecordingJobStateResponse** message with State.Sources.State parameter value not equal to "Active".
- The DUT sent **GetRecordingJobStateResponse** message with State.RecordingToken parameter value not equal to "RecordingToken1".
- The DUT sent different values for State.State in the notification message and in **GetRecordingJobStateResponse** message.

5.2.2 STOP RECORDING ON RECEIVER - PUT JOB IN IDLE STATE

Test Case ID: RECORDING-2-1-22

Specification Coverage: SetRecordingJobMode, event generation when the state field of the RecordingJobStateInformation structure is changing (ONVIF Recording Control Service Specification).

Feature Under Test: SetRecordingJobMode

WSDL Reference: recording.wsdl, receiver.wsdl, event.wsdl

Test Purpose: To verify Stop Recording on Receiver by Putting It in Idle State.

Pre-Requisite: ONVIF Client gets the entry points for Recording Control Service and Receiver Service by GetServices command. All recording jobs were stopped. Options are supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.

3. Execute [Annex A.9](#) with `RequiredSpareJobs = 0` to create or select Recording to make sure that 1 recording job can be created.
4. Execute [Annex A.10](#) for Auto creation of receiver by create recording job with Idle mode.
5. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with `tns1:RecordingConfig/JobState` Topic as Filter and an `InitialTerminationTime = TerminationTime1` to check Recording Job State changing.
6. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
7. ONVIF Client will invoke **ConfigureReceiverRequest** message (`ReceiverToken = ReceiverToken1`, `Configuration.Mode = "AlwaysConnect"`, `Configuration.MediaUri` as `stream_uri` of RTSP Simulator, `Configuration.StreamSetup.Stream = "RTP-Unicast"`, `StreamSetup.Transport.Tunnel.Protocol = "UDP"`, no `StreamSetup.Transport.Tunnel.Tunnel`) to configure the receiver to receive media from RTSP Simulator.
8. Verify **ConfigureReceiverResponse** message from the DUT.
9. ONVIF Client will invoke **GetReceiverRequest** message with `ReceiverToken = ReceiverToken1`.
10. Verify **GetReceiverResponse** message from the DUT. Check that **GetReceiverResponse** message contains the same parameters values as were changed in **ConfigureReceiverRequest** message.
11. ONVIF Client will invoke **SetRecordingJobModeRequest** message (`JobToken = "JobToken1"`, `Mode = "Active"`) to start recording.
12. Verify **SetRecordingJobModeResponse** message from the DUT. Mark time of **CreateRecordingJobResponse** as T1.
13. Execute [Annex A.5](#) for receiving event with "Active" or "PartiallyActive" Recording job state.
14. ONVIF Client will invoke **GetRecordingJobStateRequest** message with `JobToken = "JobToken1"`.
15. Verify the **GetRecordingJobStateResponse** message from the DUT. Check that `State.State = "Active"` or `"PartiallyActive"`.
16. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with `tns1:RecordingConfig/JobState` Topic as Filter and an `InitialTerminationTime = TerminationTime1` to check Recording Job State changing.
17. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.

18. ONVIF Client will invoke **SetRecordingJobModeRequest** message (JobToken = "JobToken1", Mode = "Idle") to stop recording.
19. Verify **SetRecordingJobModeResponse** message from the DUT. Remark T1 as time of **SetRecordingJobModeResponse** message.
20. Execute [Annex A.5](#) for receiving event with Idle state.
21. ONVIF Client will invoke **GetRecordingJobStateRequest** message with JobToken = "JobToken1".
22. Verify the **GetRecordingJobStateResponse** message from the DUT. Check that State.State and State.Sources.State parameter values are equal to "Idle".

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **CreatePullPointSubscriptionRequest** message
- The DUT did not send a valid **ConfigureReceiverResponse** message.
- The DUT did not send a valid **GetReceiverResponse** message.
- The DUT did not send a valid **SetRecordingJobModeResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message.
- The DUT did not send a valid **DeleteRecordingJobResponse** message.
- The DUT sent **GetReceiverResponse** message with parameters values different from sent in **ConfigureReceiverRequest** message.
- The DUT returned **GetRecordingJobStateResponse** State.State different from "Active" or "PartailActive" at step 15.
- The DUT sent **GetRecordingJobStateResponse** message with State.State parameter value not equal to "Idle" at step 22.
- The DUT sent **GetRecordingJobStateResponse** message with State.Sources.State parameter value not equal to "Idle" at step 22.
- The DUT sent **GetRecordingJobStateResponse** message with State.RecordingToken parameter value not equal to "RecordingToken1".

5.2.3 STOP RECORDING ON RECEIVER - NEVER CONNECTED MODE

Test Case ID: RECORDING-2-1-23

Specification Coverage: Event generation when the receiver state is changing (ONVIF Receiver Service Specification).

Feature Under Test: None

WSDL Reference: recording.wsdl, receiver.wsdl, event.wsdl

Test Purpose: To verify Stop Recording on Receiver by Putting Receiver in NeverConnect Mode.

Pre-Requisite: ONVIF Client gets the entry points for Recording Control Service and Receiver Service by GetServices command. All recording jobs were stopped. Options are supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.9](#) with RequiredSpareJobs = 0 to create or select Recording to make sure that 1 recording job can be created.
4. Execute [Annex A.10](#) for Auto creation of receiver by create recording job with Idle mode.
5. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:Receiver/ChangeState Topic as Filter and an InitialTerminationTime = Time1 to check Receiver state.
6. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
7. ONVIF Client will invoke **ConfigureReceiverRequest** message with ReceiverToken = ReceiverToken1 (Configuration.Mode = "AlwaysConnect", Configuration.MediaUri as stream_uri of RTSP Simulator, Configuration.StreamSetup.Stream = "RTP-Unicast", StreamSetup.Transport.Tunnel.Protocol = "UDP", no StreamSetup.Transport.Tunnel.Tunnel) to configure the receiver to receive media from RTSP Simulator.
8. Verify **ConfigureReceiverResponse** message from the DUT.

9. ONVIF Client will invoke **GetReceiverRequest** message with ReceiverToken = ReceiverToken1.
10. Verify **GetReceiverResponse** message from the DUT. Check that **GetReceiverResponse** message contains the same parameters values as were changed in **ConfigureReceiverRequest** message.
11. ONVIF Client will invoke **SetRecordingJobModeRequest** message (JobToken = "JobToken1", Mode = "Active") to start the recording.
12. Verify **SetRecordingJobModeResponse** message from the DUT. Mark the time of **SetRecordingJobModeRequest** message as T1.
13. Execute [Annex A.6](#) for receiving event with Connected state of the receiver.
14. ONVIF Client will invoke **GetReceiverStateRequest** message to check Receiver State.
15. Verify **GetReceiverStateResponse** message from the DUT. Check that ReceiverState.State = "Connected".
16. ONVIF Client will invoke **GetRecordingJobStateRequest** message with JobToken = "JobToken1".
17. Verify the **GetRecordingJobStateResponse** message from the DUT.
18. Check that State.State is equal to "Active" or "PartiallyActive".
19. Check that State.Sources.State is equal to "Active".
20. Check that State.State value in the **GetRecordingJobStateResponse** message equals to State.State value in the notification message.
21. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:Receiver/ChangeState Topic as Filter and an InitialTerminationTime = Time1 to check Receiver state.
22. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
23. ONVIF Client will invoke **SetReceiverModeRequest** message (ReceiverToken = "ReceiverToken1", Mode = "NeverConnect") to stop recording.
24. Verify **SetReceiverModeResponse** message from the DUT.
25. Execute [Annex A.6](#) for receiving event with NotConnected state of the receiver.
26. ONVIF Client will invoke **GetReceiverStateRequest** message to check Receiver State.

27. Verify **GetReceiverStateResponse** message from the DUT. Check that ReceiverState.State = "NotConnected".
28. ONVIF Client will invoke **GetRecordingJobStateRequest** message with JobToken = "JobToken1".
29. Verify the **GetRecordingJobStateResponse** message from the DUT. Check that State.State and State.Sources.State parameter values are equal to "Idle".

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **CreatePullPointSubscriptionRequest** message
- The DUT did not send a valid **ConfigureReceiverResponse** message.
- The DUT did not send a valid **GetReceiverResponse** message.
- The DUT did not send a valid **SetRecordingJobModeResponse** message.
- The DUT did not send a valid **GetReceiverStateResponse** message.
- The DUT did not send a valid **SetReceiverModeResponse** message.
- The DUT sent **GetReceiverResponse** message with parameters values differ from sent in **ConfigureReceiverRequest** message.
- The DUT sent **GetReceiverStateResponse** with ReceiverState.State not equal to "Connected" at step 15.
- The DUT returned **GetRecordingJobStateResponse** State.State different from "Active" or "PartailActive" at step 18.
- The DUT sent **GetRecordingJobStateResponse** message with State.Sources.State parameter value not equal to "Active" at step 19.
- The DUT sent **GetReceiverStateResponse** with ReceiverState.State not equal to "NotConnected" at step 27.
- The DUT sent **GetRecordingJobStateResponse** message with State.RecordingToken parameter value not equal to "RecordingToken1".

- The DUT sent **GetRecordingJobStateResponse** message with State.State and State.Sources.State parameters values not equal to "Idle" at step 29.

5.2.4 STOP RECORDING ON RECEIVER - DELETE JOB

Test Case ID: RECORDING-2-1-25

Specification Coverage: DeleteRecordingJob (ONVIF Recording Control Service Specification)

Feature Under Test: DeleteRecordingJob

WSDL Reference: recording.wsdl, receiver.wsdl, event.wsdl

Test Purpose: To verify Stop Recording on Receiver by Job Deletion.

Pre-Requisite: ONVIF Client gets the entry points for Recording Control Service and Receiver Service by GetServices command. All recording jobs were stopped. Options are supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.9](#) with RequiredSpareJobs = 0 to create or select Recording to make sure that 1 recording job can be created.
4. Execute [Annex A.10](#) for auto creation of receiver by create recording job with Idle mode.
5. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/JobState Topic as Filter and an InitialTerminationTime = TerminationTime1 to check Recording Job State changing.
6. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
7. ONVIF Client will invoke **ConfigureReceiverRequest** message (ReceiverToken = ReceiverToken1, Configuration.Mode = "AlwaysConnect", Configuration.MediaUri as stream_uri of RTSP Simulator, Configuration.StreamSetup.Stream = "RTP-Unicast", StreamSetup.Transport.Tunnel.Protocol = "UDP", no StreamSetup.Transport.Tunnel.Tunnel) to configure the receiver to receive media from RTSP Simulator.
8. Verify **ConfigureReceiverResponse** message from the DUT.

9. ONVIF Client will invoke **GetReceiverRequest** message with ReceiverToken = ReceiverToken1.
10. Verify **GetReceiverResponse** message from the DUT. Check that **GetReceiverResponse** message contains the same parameters values as were changed in **ConfigureReceiverRequest** message.
11. ONVIF Client will invoke **SetRecordingJobModeRequest** message (JobToken = "JobToken1", Mode = "Active") to start recording.
12. Verify **SetRecordingJobModeResponse** message from the DUT. Mark time of **CreateRecordingJobResponse** as T1.
13. Execute [Annex A.5](#) for receiving event with "Active" or "PartiallyActive" Recording job state.
14. ONVIF Client will invoke **GetRecordingJobStateRequest** message with JobToken = "JobToken1".
15. Verify the **GetRecordingJobStateResponse** message from the DUT. Check that State.State = "Active" or "PartiallyActive".
16. ONVIF Client will invoke **DeleteRecordingJobRequest** with JobToken = JobToken1 to delete recording job.
17. Verify **DeleteRecordingJobResponse** from the DUT.
18. ONVIF Client will invoke **GetRecordingJobsRequest** to get full list of the recording jobs.
19. Verify the **GetRecordingJobsResponse** from the DUT.
20. Check that there is no JobItem with JobToken1 in the **GetRecordingJobsResponse**.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **CreatePullPointSubscriptionRequest** message
- The DUT did not send a valid **ConfigureReceiverResponse** message.
- The DUT did not send a valid **GetReceiverResponse** message.
- The DUT did not send a valid **SetRecordingJobModeResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message.

- The DUT did not send a valid **DeleteRecordingJobResponse** message.
- The DUT did not send a valid **GetRecordingJobsResponse** message.
- The DUT sent **GetReceiverResponse** message with parameters values different from the ones sent in **ConfigureReceiverRequest** message.
- The DUT sent **GetRecordingJobStateResponse** message with State.State parameter value not equal to "Active" or "PartiallyActive" at step 15.
- The DUT returned the JobToken1 in **GetRecordingJobsResponse** message after deleting.

5.2.5 MODIFY MEDIA ATTRIBUTE WHILE RECORDING - RECEIVER

Test Case ID: RECORDING-2-1-27

Specification Coverage: Event generation when the receiver state is changing (ONVIF Receiver Service Specification).

Feature Under Test: None

WSDL Reference: recording.wSDL, receiver.wSDL, event.wSDL

Test Purpose: To verify Media Attributes Modification for Remote Recordings while Recording.

Pre-Requisite: ONVIF Client gets the entry points for Recording Control Service and Receiver Service by GetServices command. All recording jobs were stopped. Options are supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.9](#) with RequiredSpareJobs = 0 to create or select Recording to make sure that 1 recording job can be created.
4. Execute [Annex A.10](#) for auto creation of receiver by create recording job with Idle mode.
5. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/JobState Topic as Filter and an InitialTerminationTime = TerminationTime1 to check Recording Job State changing.

6. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
7. ONVIF Client will invoke **ConfigureReceiverRequest** message (ReceiverToken = ReceiverToken1, Configuration.Mode = "AlwaysConnect", Configuration.MediaUri = stream_uri1, (stream_uri1 from RTSP Simulator), Configuration.StreamSetup.Stream = "RTP-Unicast", StreamSetup.Transport.Tunnel.Protocol = "UDP", no StreamSetup.Transport.Tunnel.Tunnel) to configure the receiver to receive media from RTSP Simulator.
8. Verify **ConfigureReceiverResponse** message from the DUT.
9. ONVIF Client will invoke **GetReceiverRequest** message with ReceiverToken = ReceiverToken1.
10. Verify **GetReceiverResponse** message from the DUT. Check that **GetReceiverResponse** message contains the same parameters values as were changed in **ConfigureReceiverRequest** message.
11. ONVIF Client will invoke **SetRecordingJobModeRequest** message (JobToken = "JobToken1", Mode = "Active") to start recording.
12. Verify **SetRecordingJobModeResponse** message from the DUT. Mark time of **CreateRecordingJobResponse** as T1.
13. Execute [Annex A.5](#) for receiving event with "Active" or "PartiallyActive" Recording job state.
14. ONVIF Client will invoke **GetRecordingJobStateRequest** message with JobToken = "JobToken1".
15. Verify the **GetRecordingJobStateResponse** message from the DUT. Check that State.State = "Active" or "PartiallyActive".
16. ONVIF Client will invoke **ConfigureReceiverRequest** message (ReceiverToken = ReceiverToken1, Configuration.Mode = "AlwaysConnect", Configuration.MediaUri = stream_uri2, (where stream_uri2 is streaming of RTSP Simulator with resolution, bitrate, framerate and quality are different from stream_uri1), Configuration.StreamSetup.Stream = "RTP-Unicast", StreamSetup.Transport.Tunnel.Protocol = "UDP", no StreamSetup.Transport.Tunnel.Tunnel) to modify stream settings of the RTSP simulator.
17. Verify **ConfigureReceiverResponse** message from the DUT.
18. ONVIF Client will invoke **GetRecordingJobStateRequest** message with JobToken = "JobToken1".
19. Verify the **GetRecordingJobStateResponse** message from the DUT. Check that State.State = "Active" or "PartiallyActive".

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **CreatePullPointSubscriptionRequest** message
- The DUT did not send a valid **ConfigureReceiverResponse** message.
- The DUT did not send a valid **GetReceiverResponse** message.
- The DUT did not send a valid **SetRecordingJobModeResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message.
- The DUT returned Recording parameters in **GetRecordingsResponse** message that differ from specified during Recording changing.
- The DUT sent **GetRecordingJobStateResponse** message with State.State parameter value not equal to "Active" or "PartiallyActive" at step 15 or 19.
- The DUT sent **GetRecordingJobStateResponse** message with State.RecordingToken parameter value not equal to "RecordingToken1".

5.2.6 START RECORDING ON MEDIA PROFILE

Test Case ID: RECORDING-2-1-28

Specification Coverage: Creation of a Recording Job with Active mode on local storage. Event generation when the state field of the RecordingJobStateInformation structure is changing (ONVIF Recording Control Service Specification).

Feature Under Test: CreateRecordingJob

WSDL Reference: recording.wsdl, media.wsdl, event.wsdl

Test Purpose: To verify Start Recording using a Media Profile.

Pre-Requisite: ONVIF Client gets the entry points for Recording Control Service and Media Service by GetServices command. At least one media profile compatible with at least one existing or created recording exists on the DUT. All recording jobs were stopped. Options are supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.12](#) with `RequiredSpareJobs = 0` to create or select Recording (`RecordingToken = RecordingToken1`) to make sure that 1 recording job can be created and to get the compatible media profile tokens list.
4. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with `tns1:RecordingConfig/JobState` Topic as Filter and an `InitialTerminationTime = TerminationTime1` to check Recording Job State changing.
5. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
6. ONVIF Client will invoke **CreateRecordingJobRequest** message (`JobConfiguration.RecordingToken = RecordingToken1`, `JobConfiguration.Mode = Active`, `JobConfiguration.Priority = 1`, `JobConfiguration.Source.SourceToken.Token = "ProfileToken1"`, where `ProfileToken1` is token of `MediaProfile` from `Compatible Sources` list, `JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Profile"`, `JobConfiguration.Source.AutoCreateReceiver` is not present) to create a recording job for configured recording with Active mode.
7. Verify the **CreateRecordingJobResponse** message (`JobToken = JobToken1`).
8. Verify that `JobConfiguration` in **CreateRecordingJobResponse** message contains the same parameters values as was sent in **CreateRecordingJobRequest** message.
9. Execute [Annex A.13](#) for receiving `Initialized` event with "Active" or "PartiallyActive" Recording job state.
10. ONVIF Client will invoke **GetRecordingJobStateRequest** message with `JobToken = "JobToken1"`.
11. Verify the **GetRecordingJobStateResponse** message from the DUT.
12. Check that `State.State` is equal to "Active" or "PartiallyActive".
13. Check that `State.Sources.State` is equal to "Active".
14. Check that `State.State` value in the **GetRecordingJobStateResponse** message equals to `State.State` value in the notification message.

Test Result:

PASS –

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **CreatePullPointSubscriptionResponse** message.
- The DUT did not send a valid **CreateRecordingJobResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message.
- The DUT sent **CreateRecordingJobResponse** message with JobConfiguration parameters values differ from sent in **CreateRecordingJobRequest** message.
- The DUT sent **GetRecordingJobStateResponse** message with State.State parameter value not equal to "Active" or "PartiallyActive".
- The DUT sent **GetRecordingJobStateResponse** message with State.Sources.State parameter value not equal to "Active".
- The DUT sent **GetRecordingJobStateResponse** message with State.RecordingToken parameter value not equal to "RecordingToken1".
- The DUT sent different values for State.State in the notification message and in **GetRecordingJobStateResponse** message.

5.2.7 STOP RECORDING ON MEDIA PROFILE - PUT JOB IN IDLE STATE

Test Case ID: RECORDING-2-1-29

Specification Coverage: SetRecordingJobMode, event generation when the state field of the RecordingJobStateInformation structure is changing (ONVIF Recording Control Service Specification).

Feature Under Test: SetRecordingJobMode

WSDL Reference: recording.wsdl, media.wsdl, event.wsdl

Test Purpose: To verify Stop Recording using a Media Profile by Putting It in Idle State.

Pre-Requisite: ONVIF Client gets the entry points for Recording Control Service and Media Service by GetServices command. At least one media profile is compatible with at least one existing or created recording exists on the DUT. All recording jobs were stopped. Options are supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.12](#) with `RequiredSpareJobs = 0` to create or select Recording (`RecordingToken = RecordingToken1`) to make sure that 1 recording job can be created and to get the compatible media profile tokens list.
4. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with `tns1:RecordingConfig/JobState` Topic as Filter and an `InitialTerminationTime = TerminationTime1` to check Recording Job State changing.
5. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
6. ONVIF Client will invoke **CreateRecordingJobRequest** message (`JobConfiguration.RecordingToken = RecordingToken1`, `JobConfiguration.Mode = Active`, `JobConfiguration.Priority = 1`, `JobConfiguration.Source.SourceToken.Token = "ProfileToken1"`, where `ProfileToken1` is token of `MediaProfile` from `Compatible Sources` list, `JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Profile"`, `JobConfiguration.Source.AutoCreateReceiver` is not present) to create a recording job for configured recording with Active mode.
7. Verify the **CreateRecordingJobResponse** message (`JobToken = JobToken1`).
8. Verify that `JobConfiguration` in **CreateRecordingJobResponse** message contains the same parameters values as was sent in **CreateRecordingJobRequest** message.
9. Execute [Annex A.13](#) for receiving `Initialized` event with "Active" or "PartiallyActive" Recording job state.
10. ONVIF Client will invoke **GetRecordingJobStateRequest** message with `JobToken = "JobToken1"`.
11. Verify the **GetRecordingJobStateResponse** message from the DUT. Check that `State.State` parameter value is equal to "Active" or "PartiallyActive".
12. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with `tns1:RecordingConfig/JobState` Topic as Filter and an `InitialTerminationTime = TerminationTime1` to check Recording Job State changing.
13. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.

14. ONVIF Client will invoke **SetRecordingJobModeRequest** message (JobToken = "JobToken1", Mode = "Idle") to stop recording.
15. Verify **SetRecordingJobModeResponse** message from the DUT. Remark Time1 as time of receiving of **SetRecordingJobModeResponse** message
16. Execute [Annex A.5](#) for receiving changed event with "Idle" Recording job state.
17. ONVIF Client will invoke **GetRecordingJobStateRequest** message with JobToken = "JobToken1".
18. Verify the **GetRecordingJobStateResponse** message from the DUT. Check that State.State and State.Sources.State parameter values are equal to "Idle".

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **CreatePullPointSubscriptionRequest** message
- The DUT did not send a valid **CreateRecordingJobResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message.
- The DUT did not send a valid **SetRecordingJobModeResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message
- The DUT returned JobConfiguration parameters in **CreateRecordingJobResponse** message that differs from specified during RecordingJob creation.
- The DUT returned **GetRecordingJobStateResponse** State.State differs from "Active" or "PartiallyActive" at step 11.
- The DUT sent **GetRecordingJobStateResponse** message with State.State parameter value not equal to "Idle" at step 18.
- The DUT sent **GetRecordingJobStateResponse** message with State.Sources.State parameter value not equal to "Idle" at the step 18.
- The DUT sent **GetRecordingJobStateResponse** message with State.RecordingToken parameter value not equal to "RecordingToken1".

5.2.8 STOP RECORDING ON MEDIA PROFILE - DELETE JOB

Test Case ID: RECORDING-2-1-30

Specification Coverage: DeleteRecordingJob (ONVIF Recording Control Service Specification)

Feature Under Test: DeleteRecordingJob

WSDL Reference: recording.wsdl, media.wsdl, event.wsdl

Test Purpose: To verify Stop Recording using a Media Profile by Job Deletion.

Pre-Requirement: ONVIF Client gets the entry points for Recording Control Service and Media Service by GetServices command. All recording jobs were stopped. At least one media profile compatible with at least one existing or created recording exists on the DUT. Options are supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.12](#) with RequiredSpareJobs = 0 to create or select Recording (RecordingToken = RecordingToken1) to make sure that 1 recording job can be created and to get the compatible media profile tokens list.
4. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/JobState Topic as Filter and an InitialTerminationTime = TerminationTime1 to check Recording Job State changing.
5. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
6. ONVIF Client will invoke **CreateRecordingJobRequest** message (JobConfiguration.RecordingToken = RecordingToken1, JobConfiguration.Mode = Active, JobConfiguration.Priority = 1, JobConfiguration.Source.SourceToken.Token = "ProfileToken1", where ProfileToken1 is token of MediaProfile from Compatible Sources list, JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Profile", JobConfiguration.Source.AutoCreateReceiver is not present) to create a recording job for configured recording with Active mode.
7. Verify the **CreateRecordingJobResponse** message (JobToken = JobToken1).

8. Verify that JobConfiguration in **CreateRecordingJobResponse** message contains the same parameters values as was sent in **CreateRecordingJobRequest** message.
9. Execute [Annex A.13](#) for receiving Initialized event with "Active" or "PartiallyActive" Recording job state.
10. ONVIF Client will invoke **GetRecordingJobStateRequest** message with JobToken = "JobToken1".
11. Verify the **GetRecordingJobStateResponse** message from the DUT. Check that State.State and State.Sources.State parameter values are equal to "Active" or "PartiallyActive".
12. ONVIF Client will invoke **DeleteRecordingJobRequest** message (JobToken = JobToken1) to delete Recording Job.
13. Verify the **DeleteRecordingJobResponse** message from the DUT.
14. ONVIF Client will invoke **GetRecordingJobsRequest** to get full list of the recording jobs.
15. Verify the **GetRecordingJobsResponse** from the DUT.
16. Check that there is no JobItem with JobToken1 in the **GetRecordingJobsResponse**.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **CreatePullPointSubscriptionRequest** message
- The DUT did not send a valid **CreateRecordingJobResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message.
- The DUT did not send a valid **GetRecordingJobsResponse** message.
- The DUT sent **CreateRecordingJobResponse** message with JobConfiguration parameters values different from sent in **CreateRecordingJobRequest** message.
- The DUT sent **GetRecordingJobStateResponse** message with State.State parameter value not equal to "Active" or "PartiallyActive".
- The DUT returned the JobToken1 in **GetRecordingJobsResponse** message after deleting.

5.3 Configuration

5.3.1 DYNAMIC TRACKS CONFIGURATION

Test Case ID: RECORDING-3-1-7

Specification Coverage: CreateTrack, DeleteTrack (ONVIF Recording Control Service Specification).

Feature Under Test: CreateTrack, DeleteTrack

WSDL Reference: recording.wsdl

Test Purpose: To verify creation and deletion of tracks.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices command. Dynamic Tracks functionality is supported by the DUT. Options are supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.11](#) for selection of recording to make sure that track creation procedure will be possible.
4. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve initial track list of the recordings.
5. Verify the **GetRecordingsResponse** message from the DUT.
6. ONVIF Client will invoke **CreateTrackRequest** message (RecordingToken=RecordingToken1 where RecordingToken1 is RecordingToken from [Annex A.11](#), TrackConfiguration.TrackType as track type from [Annex A.11](#), TrackConfiguration.Description = 'New Track') to create a new track in the recording.
7. Verify the **CreateTrackResponse** message (TrackToken = TrackToken1) from the DUT.
8. Check that TrackToken = TrackToken1 from **CreateTrackResponse** message was not presented in Recording with token RecordingToken1 in initial track list in **GetRecordingsResponse** message at step 5.

9. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve updated track list of the recordings.
10. Verify the **GetRecordingsResponse** message from the DUT.
11. Check that new track (TrackToken = TrackToken1) is presented in updated track list for RecordingToken = RecordingToken1. Check that all parameter values are the same as were sent in **CreateTrackRequest** message.
12. Check that all other tracks for RecordingToken = RecordingToken1 in **GetRecordingsResponse** message at step 10 have the same parameters as were in **GetRecordingsResponse** message at step 5. Check that all other recordings have the same parameters value as were in **GetRecordingsResponse** message at step 5.
13. ONVIF Client will invoke **DeleteTrackRequest** message (RecordingToken = RecordingToken1, TrackToken = TrackToken1) to delete the track.
14. Verify the **DeleteTrackResponse** message from the DUT.
15. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve updated track list.
16. Verify the **GetRecordingsResponse** message from the DUT. Check that track (TrackToken = TrackToken1) for RecordingToken = RecordingToken1 is no longer present on track list.
17. Check that all other tracks in **GetRecordingsResponse** message at step 16 have the same parameters as were in **GetRecordingsResponse** message at step 5. Check that all other recordings have the same parameters value as were in **GetRecordingsResponse** message at step 5.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetRecordingsResponse** message.
- The DUT did not send a valid **CreateTrackResponse** message.
- The DUT did not send a valid **DeleteTrackResponse** message.
- The DUT returned TrackToken value in the **CreateTrackResponse** not unique with Track Tokens for Recording with RecordingToken = RecordingToken1 in initial **GetRecordingsResponse** message at step 5.

- The DUT did not return TrackToken = TrackToken1 for Recording with RecordingToken = RecordingToken1 in **GetRecordingsResponse** message after track creation.
- The parameter values of the TrackToken = TrackToken1 for recording (RecordingToken = RecordingToken1) in updated recordings list at step 10 differ from values in **CreateTrackRequest** message.
- The DUT returned TrackToken = TrackToken1 for Recording with RecordingToken = RecordingToken1 in **GetRecordingsResponse** message after recording deletion.
- Any changes were applied to tracks of other recordings from initial **GetRecordingsResponse** message list during the test process.
- Any changes were applied to other tracks of recordings with RecordingToken = RecordingToken1 from initial **GetRecordingsResponse** message list during the test process.
- The DUT did not send Recording with RecordingToken = RecordingToken1 in updated RecordingList
- The DUT will not allow creating new Track for selected Recording

Note: See [Annex A.8](#) for Recording Source Information Parameters Length limitations.

Note: DUT shall provide Recording with Options.Track.SpareTotal > 0 to avoid track's data deletion.

5.3.2 DYNAMIC RECORDINGS CONFIGURATION

Test Case ID: RECORDING-3-1-10

Specification Coverage: CreateRecording, DeleteRecording

Feature Under Test: CreateRecording, DeleteRecording (ONVIF Recording Control Service Specification)

WSDL Reference: recording.wsdl

Test Purpose: To verify creation and deletion of recordings.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices command. Dynamic Recording functionality is supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.
3. Execute [Annex A.7](#) for possibility to create a new recording.
4. ONVIF Client will invoke **GetServiceCapabilitiesRequest** message to retrieve Encoding list and MetadataRecording capabilities of the Recording service of the DUT.
5. Verify **GetServiceCapabilitiesResponse** message from the DUT.
6. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve initial complete recordings list.
7. Verify the **GetRecordingsResponse** message from the DUT.
8. ONVIF Client will invoke **CreateRecordingRequest** message with RecordingConfiguration.Source.SourceId as any URI, RecordingConfiguration.Source.Name = "CameraName", RecordingConfiguration.Source.Location = "LocationDescription", RecordingConfiguration.Source.Description = "Source Description", RecordingConfiguration.Source.Address as address of the device, RecordingConfiguration.Content = "Create/Delete recordings test", RecordingConfiguration.MaximumRetentionTime = "PT0S" to create a new recording.
9. Verify the **CreateRecordingResponse** message from the DUT (RecordingToken = RecordingToken1).
10. Check that RecordingToken = RecordingToken1 from **CreateRecordingResponse** message was not presented in initial recordings list in the **GetRecordingsResponse** message.
11. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve updated recordings list.
12. Verify the **GetRecordingsResponse** message from the DUT.
13. Check that created recording (RecordingToken = RecordingToken1) is present on the updated recordings list. Check that all parameter values are the same as were sent in **CreateRecordingRequest** message.
14. If Capabilities.Encoding in **GetServiceCapabilitiesResponse** message received from the DUT at step 5 contains at least one enumeration value of tt:VideoEncoding, then ONVIF Client checks that created recording contains at least one track with track type equals to Video.
15. If Capabilities.Encoding in **GetServiceCapabilitiesResponse** message received from the DUT at step 5 contains at least one enumeration value of tt:AudioEncoding, then ONVIF

- Client checks that created recording contains at least one track with track type equals to Audio.
16. If MetadataRecording is supported by the DUT, check that created recording contains track with track type equals to Metadata.
 17. Check that all other recordings in **GetRecordingsResponse** message at step 12 have the same settings as were in **GetRecordingsResponse** message at step 7.
 18. ONVIF Client will invoke **DeleteRecordingRequest** message (RecordingToken = RecordingToken1) to delete recording.
 19. Verify the **DeleteRecordingResponse** message from the DUT.
 20. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve updated recordings list.
 21. Verify the **GetRecordingsResponse** message from the DUT. Check that recording (RecordingToken = RecordingToken1) is no longer present on recordings list.
 22. Check that all other recordings in **GetRecordingsResponse** message at step 20 have the same settings as the ones in **GetRecordingsResponse** message at step 7.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetRecordingsResponse** message.
- The DUT did not send a valid **CreateRecordingResponse** message.
- The DUT did not send a valid **DeleteRecordingResponse** message.
- The DUT did not send a valid **GetServiceCapabilitiesResponse** message.
- The DUT returned RecordingToken value in the **CreateRecordingResponse** not unique with Recording Tokens in initial **GetRecordingsResponse** message at step 7.
- The DUT did not return Recording with RecordingToken = RecordingToken1 in **GetRecordingsResponse** message after recording creation at step 12.
- The parameter values of recording (RecordingToken = RecordingToken1) on the updated recordings list at step 12 differ from values in **CreateRecordingRequest** message.

- Recording (RecordingToken = RecordingToken1) in updated recordings list at step 12 did not contain track with track type equal to 'Video' if encoding list contained at least one enumeration value of tt:VideoEncoding.
- Recording (RecordingToken = RecordingToken1) on the updated recordings list at step 12 did not contain track with track type equal to 'Audio' if encoding list contained at least one enumeration value of tt:AudioEncoding.
- Recording (RecordingToken = RecordingToken1) on the updated recordings list at step 12 did not contain track with track type equals to Metadata if MetadataRecording is supported.
- The DUT returned Recording with RecordingToken = RecordingToken1 in **GetRecordingsResponse** message after recording deletion at step 20
- Any changes were applied to other recordings from initial **GetRecordingsResponse** message list during the test process – at step 12 and 21.

Note: See [Annex A.8](#) for Recording Source Information Parameters Length limitations.

5.3.3 RECORDING JOB CONFIGURATION - DIFFERENT PRIORITIES (ON MEDIA PROFILE)

Test Case ID: RECORDING-3-1-11

Specification Coverage: Priority scheme for recording jobs (ONVIF Recording Control Service Specification).

Feature Under Test: None

WSDL Reference: recording.wsdl

Test Purpose: To verify creation of recording jobs with different priorities.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices command. No active recording jobs. At least one media profile compatible with at least one existing or created recording exists on the DUT. Options are supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.

3. Execute [Annex A.12](#) with `RequiredSpareJobs = 1` to create or select Recording (`RecordingToken = RecordingToken1`) to make sure that 2 recording jobs can be created and to get the compatible media profile tokens list.
4. ONVIF Client will invoke **CreateRecordingJobRequest** message (`JobConfiguration.RecordingToken = RecordingToken1`, `JobConfiguration.Mode = Active`, `JobConfiguration.Priority = 1`, `JobConfiguration.Source.SourceToken.Token = "ProfileToken1"`, where `ProfileToken1` is token of `MediaProfile` from `Compatible Sources` list, `JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Profile"`, `JobConfiguration.Source.AutoCreateReceiver` is not present) to create a recording job for configured recording with Active mode.
5. Verify the **CreateRecordingJobResponse** message (`JobToken = JobToken1`).
6. Wait for `JobState` to get changed (use `Operation Delay Time`).
7. ONVIF Client will invoke **GetRecordingJobStateRequest** message (`JobToken = JobToken1`) to get actual recording job state.
8. Verify the **GetRecordingJobStateResponse** message from the DUT.
9. Check that `State.State` parameter value is "Active" or "PartiallyActive".
10. ONVIF Client will invoke **CreateRecordingJobRequest** message (`JobConfiguration.RecordingToken = RecordingToken1`, `JobConfiguration.Mode = Active`, `JobConfiguration.Priority = 2`, `JobConfiguration.Source.SourceToken.Token = "ProfileToken1"`, where `ProfileToken1` is token of `MediaProfile` configured for recording, `JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Profile"`, `JobConfiguration.Source.AutoCreateReceiver` is not present) to create a new recording job for configured recording with Active mode and higher priority.
11. Verify the **CreateRecordingJobResponse** message (`JobToken = JobToken2`).
12. Wait for `JobState` to get changed (use `Operation Delay Time`).
13. ONVIF Client will invoke **GetRecordingJobStateRequest** message (`JobToken = JobToken1`) to get actual recording job state.
14. Verify the **GetRecordingJobStateResponse** message from the DUT.
15. Check that `State.State` parameter value is "Idle".
16. ONVIF Client will invoke **GetRecordingJobStateRequest** message (`JobToken = JobToken2`) to get actual recording job state.
17. Verify the **GetRecordingJobStateResponse** message from the DUT.

18. Check that State.State parameter value is "Active" or "PartiallyActive".
19. ONVIF Client will invoke **DeleteRecordingJobRequest** message (JobToken = JobToken2) to delete Recording Job.
20. Verify the **DeleteRecordingJobResponse** message from the DUT.
21. Wait for JobState to get changed (use Operation Delay Time).
22. ONVIF Client will invoke **GetRecordingJobStateRequest** message (JobToken = JobToken1) to get actual recording job state.
23. Verify the **GetRecordingJobStateResponse** message.
24. Check that State.State parameter value is "Active" or "PartiallyActive". Check that State.State value equals to State.State value at step 8.
25. ONVIF Client will invoke **DeleteRecordingJobRequest** message (JobToken = JobToken1) to delete Recording Job.
26. Verify the **DeleteRecordingJobResponse** message from the DUT.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **CreateRecordingJobResponse** message.
- The DUT did not send a valid **GetRecordingOptionsResponse** message.
- The DUT did not send a valid **GetRecordingJobsResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message.
- The DUT did not send a valid **DeleteRecordingJobResponse** message.
- The DUT returned RecordingJob parameters in **CreateRecordingJobResponse** message that differ from specified during RecordingJob creation.
- The DUT returned incorrect RecordingJobState in **GetRecordingJobStateResponse** message at steps 8, 14, 17 or 23.
- The DUT sent **GetRecordingJobStateResponse** at step 23 with State.State value different from State.State value at step 8.

5.3.4 RECORDING JOB CONFIGURATION - DIFFERENT PRIORITIES (ON RECEIVER)

Test Case ID: RECORDING-3-1-12

Specification Coverage: Priority scheme for recording jobs (ONVIF Recording Control Service Specification).

Feature Under Test: None

WSDL Reference: recording.wsdl

Test Purpose: To verify creation of recording jobs with different priorities.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices command. There are no active recording jobs. Options are supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.9](#) with RequiredSpareJobs = 1 to create or select Recording to make sure that 2 recording jobs can be created.
4. Execute [Annex A.10](#) for auto creation of receiver by create recording job with Idle mode.
5. ONVIF Client will invoke **SetRecordingJobModeRequest** message (JobToken = "JobToken1", Mode = "Active") to start recording.
6. Verify **SetRecordingJobModeResponse** message from the DUT.
7. Wait for JobState to get changed (use Operation Delay Time).
8. ONVIF Client will invoke **GetRecordingJobStateRequest** (JobToken = "JobToken1") request message to retrieve recording job state.
9. Verify the **GetRecordingJobStateResponse** from the DUT.
10. Check that State.State parameter value is equal to "Active" or "PartiallyActive".
11. ONVIF Client will invoke **CreateRecordingJobRequest** message (JobConfiguration.RecordingToken = "RecordingToken1", JobConfiguration.Mode = "Active", JobConfiguration.Priority = 2, JobConfiguration.Source.SourceToken.Token =

- "ReceiverToken1", JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Receiver", JobConfiguration.Source.AutoCreateReceiver = false) to create a recording job for the same receiver with higher priority.
12. Verify the **CreateRecordingJobResponse** message (JobToken = JobToken2).
 13. Wait for JobState to get changed (use Operation Delay Time).
 14. ONVIF Client will invoke **GetRecordingJobStateRequest** (JobToken = JobToken1) request message to retrieve recording job state.
 15. Verify the **GetRecordingJobStateResponse** from the DUT.
 16. Check that State.State parameter value is equal to "Idle".
 17. ONVIF Client will invoke **GetRecordingJobStateRequest** (JobToken = JobToken2) request message to retrieve recording job state.
 18. Verify the **GetRecordingJobStateResponse** from the DUT.
 19. Check that State.State parameter value is equal to "Active" or "PartiallyActive".
 20. ONVIF Client will invoke **DeleteRecordingJobRequest** message (JobToken = JobToken2) to delete Recording Job.
 21. Verify the **DeleteRecordingJobResponse** message from the DUT.
 22. Wait for JobState to get changed (use Operation Delay Time).
 23. ONVIF Client will invoke **GetRecordingJobStateRequest** (JobToken = JobToken1) request message to retrieve recording job state.
 24. Verify the **GetRecordingJobStateResponse** from the DUT.
 25. Check that State.State parameter value is equal to "Active" or "PartiallyActive". Check that State.State value equals to State.State value at step 9.
 26. ONVIF Client will invoke **DeleteRecordingJobRequest** message (JobToken = JobToken1) to delete Recording Job.
 27. Verify the **DeleteRecordingJobResponse** message from the DUT.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **ConfigureReceiverResponse** message.
- The DUT did not send a valid **GetRecordingOptionsResponse** message.
- The DUT did not send a valid **CreateRecordingJobResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message.
- The DUT did not send a valid **DeleteRecordingJobResponse** message.
- The DUT returned incorrect RecordingJobState in **GetRecordingJobStateResponse** message at steps 9, 15, 18, 24.
- The DUT sent **GetRecordingJobStateResponse** at step 23 with State.State value different from State.State value at step 9.

5.4 General

5.4.1 GET RECORDINGS

Test Case ID: RECORDING-4-1-1

Specification Coverage: GetRecordings (ONVIF Recording Control Service Specification)

Feature Under Test: GetRecordings

WSDL Reference: recording.wsdl

Test Purpose: To verify Get Recordings.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command. At least one recording exists for the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.
4. Verify the **GetRecordingsResponse** message from the DUT.

Test Result:

PASS –

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetRecordingsResponse** message.
- The DUT sent at least two Recordings with the same RecordingToken in the **GetRecordingsResponse** message.

5.4.2 GET RECORDING CONFIGURATION

Test Case ID: RECORDING-4-1-2

Specification Coverage: GetRecordingConfiguration (ONVIF Recording Control Service Specification)

Feature Under Test: GetRecordingConfiguration

WSDL Reference: recording.wsdl

Test Purpose: To verify Get Recording Configuration.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.
4. Verify the **GetRecordingsResponse** message from the DUT.
5. ONVIF Client will invoke **GetRecordingConfigurationRequest** message (RecordingToken = "Token1", where Token1 is the first RecordingItem.RecordingToken from the **GetRecordingsResponse** message) to retrieve recording configuration.
6. Verify the **GetRecordingConfigurationResponse** message from the DUT.
7. Repeat steps 5-6 for all other recordings from the **GetRecordingsResponse** message.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetRecordingsResponse** message.
- The DUT did not send a valid **GetRecordingConfigurationResponse** message.
- The DUT returned different parameter values for the same recording in **GetRecordingsResponse** message and in **GetRecordingConfigurationResponse** message (see [Annex A.1](#)).

Note: If the DUT does not return any recordings in **GetRecordingsResponse** message, skip steps 5-7 and go to the next test.

Note: Two recordings are supposed to be the same, if they have equal tokens.

5.4.3 GET RECORDING CONFIGURATION WITH INVALID TOKEN

Test Case ID: RECORDING-4-1-3

Specification Coverage: GetRecordingConfiguration (ONVIF Recording Control Service Specification)

Feature Under Test: GetRecordingConfiguration

WSDL Reference: recording.wsdl

Test Purpose: To verify Get Recording Configuration with invalid Token.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.

4. Verify that the **GetRecordingsResponse** message contains at least one Recording.
5. If there is at least one Recording, then skip steps 6-7 and go to the step 8.
6. ONVIF Client will invoke **CreateRecordingRequest** message to create new Recording.
7. Verify the **CreateRecordingResponse** message from the DUT.
8. ONVIF Client will invoke **GetRecordingConfigurationRequest** message (invalid RecordingToken).
9. The DUT will generate SOAP 1.2 fault message (InvalidArgVal/NoRecording).
10. Restore DUT settings.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetRecordingsResponse** message.
- The DUT did not send a valid **CreateRecordingResponse** message.
- The DUT did not send SOAP 1.2 fault message.
- The DUT sent incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

Note: Other faults than specified in the test are acceptable, though the specified are preferable.

Note: See [Annex A.8](#) for Recording Source Information Parameters Length limitations.

5.4.4 GET RECORDING JOBS

Test Case ID: RECORDING-4-1-4

Specification Coverage: GetRecordingJobs (ONVIF Recording Control Service Specification)

Feature Under Test: GetRecordingJobs

WSDL Reference: recording.wsdl

Test Purpose: To verify Get Recording Jobs.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingJobsRequest** message to retrieve a complete recording jobs list.
4. Verify the **GetRecordingJobsResponse** message from the DUT.

Test Result:

PASS –

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetRecordingJobsResponse** message.
- The DUT sent at least two Recording Jobs with the same token in the **GetRecordingJobsResponse** message.

5.4.5 GET RECORDING JOB CONFIGURATION

Test Case ID: RECORDING-4-1-5

Specification Coverage: GetRecordingJobConfiguration (ONVIF Recording Control Service Specification)

Feature Under Test: GetRecordingJobConfiguration

WSDL Reference: recording.wsdl

Test Purpose: To verify Get Recording Job Configuration.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.

3. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.
4. Verify the **GetRecordingsResponse** message from the DUT.
5. ONVIF Client will invoke **GetRecordingJobsRequest** message to retrieve a complete recording jobs list.
6. Verify the **GetRecordingJobsResponse** message from the DUT.
7. ONVIF Client will invoke **GetRecordingJobConfigurationRequest** message (JobToken = "Token1", where Token1 is the first JobItem.JobToken from the **GetRecordingJobsResponse** message) to retrieve recording job configuration.
8. Verify the **GetRecordingJobConfigurationResponse** message from the DUT.
9. Repeat steps 7-8 for all other recording jobs from the **GetRecordingJobsResponse** message.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetRecordingsResponse** message.
- The DUT did not send a valid **GetRecordingJobsResponse** message.
- The DUT did not send a valid **GetRecordingJobConfigurationResponse** message.
- The DUT returned different parameter values for the same recording job in **GetRecordingJobsResponse** message and in **GetRecordingJobConfigurationResponse** message (see [Annex A.2](#)).
- The DUT returned JobItem.JobConfiguration.RecordingToken that does not exist in **GetRecordingsResponse** message.

Note: If the DUT does not return any recording in **GetRecordingsResponse** message, skip steps 5-9 and go to the next test.

Note: If the DUT does not return any recording job in **GetRecordingJobsResponse** message, skip steps 7-9 and go to the next test.

Note: Two recording jobs supposed to be the same, if they have equal tokens.

5.4.6 GET RECORDING JOB STATE

Test Case ID: RECORDING-4-1-7

Specification Coverage: GetRecordingJobState (ONVIF Recording Control Service Specification)

Feature Under Test: GetRecordingJobState

WSDL Reference: recording.wsdl

Test Purpose: To verify Get Recording Job State.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.
4. Verify the **GetRecordingsResponse** message from the DUT.
5. ONVIF Client will invoke **GetRecordingJobsRequest** message to retrieve a complete recording jobs list.
6. Verify the **GetRecordingJobsResponse** message from the DUT.
7. ONVIF Client will invoke **GetRecordingJobStateRequest** message (JobToken = "Token1", where Token1 is the first JobItem.JobToken from the **GetRecordingJobsResponse** message) to retrieve recording job state.
8. Verify the **GetRecordingJobStateResponse** message from the DUT.
9. Repeat steps 7-8 for all other recording jobs from the **GetRecordingJobsResponse** message.

Test Result:

PASS –

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetRecordingsResponse** message.
- The DUT did not send a valid **GetRecordingJobsResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message.
- The DUT returned different parameter values for the same recording job in **GetRecordingJobsResponse** message and in **GetRecordingJobStateResponse** message (see [Annex A.3](#)).
- The DUT returned JobItem.JobConfiguration.RecordingToken that does not exist in **GetRecordingsResponse** message.

Note: If the DUT does not return any recording in **GetRecordingsResponse** message, skip steps 5-9 and go to the next test.

Note: If the DUT does not return any recording job in **GetRecordingJobsResponse** message, skip steps 7-9 and go to the next test.

Note: Two recording jobs supposed to be the same, if they have equal tokens.

5.4.7 GET TRACK CONFIGURATION

Test Case ID: RECORDING-4-1-9

Specification Coverage: GetTrackConfiguration (ONVIF Recording Control Service Specification)

Feature Under Test: GetTrackConfiguration

WSDL Reference: recording.wsdl

Test Purpose: To verify Get Track Configuration.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.

4. Verify the **GetRecordingsResponse** message from the DUT.
5. ONVIF Client will invoke **GetTrackConfigurationRequest** message (RecordingToken = "RecordingToken1", where RecordingToken1 is the first RecordingItem.RecordingToken from the **GetRecordingsResponse** message; TrackToken = "TrackToken1", where TrackToken1 is the first RecordingItem.Tracks.Track.TrackToken for this RecordingItem from the **GetRecordingsResponse** message) to retrieve the configuration for a specific track.
6. Verify the **GetTrackConfigurationResponse** message from the DUT.
7. Repeat steps 5-6 for all other recording tracks from the **GetRecordingsResponse** message for RecordingToken1
8. Repeat steps 5-7 for all other recording tokens from the **GetRecordingsResponse** message

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetRecordingsResponse** message.
- The DUT did not send a valid **GetTrackConfigurationResponse** message.
- The DUT returned different parameter values for the same track in **GetTrackConfigurationResponse** message and in **GetRecordingsResponse** message (see [Annex A.4](#)).

Note: If the DUT does not return any tracks in **GetRecordingsResponse** message, skip steps 5-8 and go to the next test.

Note: If the DUT does not return any recording in **GetRecordingsResponse** message, skip steps 5-8 and go to the next test.

Note: Two tracks supposed to be the same, if they have equal tokens and placed in one recording.

5.4.8 GET TRACK CONFIGURATION WITH INVALID TOKEN

Test Case ID: RECORDING-4-1-10

Specification Coverage: GetTrackConfiguration (ONVIF Recording Control Service Specification)

Feature Under Test: GetTrackConfiguration

WSDL Reference: recording.wsdl

Test Purpose: To verify Get Track Configuration with invalid Token.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.
4. Verify that the **GetRecordingsResponse** message contains at least one Recording.
5. If there is at least one Recording, then skip steps 6-9 and go to the step 10.
6. ONVIF Client will invoke **CreateRecordingRequest** message to create new Recording.
7. Verify the **CreateRecordingResponse** message from the DUT.
8. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve TrackToken of created recording.
9. Verify **GetRecordingsResponse** message from the DUT (TrackToken = TrackToken1).
10. ONVIF Client will invoke **GetTrackConfigurationRequest** message (invalid TrackToken).
11. The DUT will generate SOAP 1.2 fault message (InvalidArgVal/ NoTrack).
12. ONVIF Client will invoke **GetTrackConfigurationRequest** message (invalid RecordingToken).
13. The DUT will generate SOAP 1.2 fault message (InvalidArgVal/ NoRecording).
14. Restore DUT settings.

Test Result:

PASS –

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetRecordingsResponse** message.
- The DUT did not send a valid **GetRecordingJobsResponse** message.
- The DUT did not send a valid **CreateRecordingResponse** message.
- The DUT did not send SOAP 1.2 fault message.
- The DUT sent an incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

Note: Other faults than specified in the test are acceptable, though the specified are preferable.

Note: Skip the test in case of fault in **CreateRecordingResponse** message.

Note: See [Annex A.8](#) for Recording Source Information Parameters Length limitations.

5.4.9 SET RECORDINGS CONFIGURATION (MAXIMUM LENGTH OF RECORDING SOURCE INFORMATION)

Test Case ID: RECORDING-4-1-11

Specification Coverage: Recording Source Information & wsdl (ONVIF Recording Control Service Specification)

Feature Under Test: GetRecordingConfiguration, SetRecordingConfiguration

WSDL Reference: recording.wsdl

Test Purpose: To verify applying of recording source information (Name, Address, SourceId) with maximum length by the DUT.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command. At least one recording exists for a DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.
4. Verify the **GetRecordingsResponse** message from the DUT. Select one of the recordings to use it in the further steps.

5. ONVIF Client will invoke **SetRecordingConfigurationRequest** message (RecordingToken = [selected recording], RecordingConfiguration.Source.Name = [string with length 20 characters, which contains only readable characters and begins with letter], RecordingConfiguration.Source.SourceId = [valid URI with length 128 characters], RecordingConfiguration.Source.Address = [valid URI with length 128 characters], other Recording Configuration parameters without change) to configure Recording.
6. Verify the **SetRecordingConfigurationResponse** message from the DUT.
7. ONVIF Client will invoke **GetRecordingConfigurationRequest** message for selected Recording.
8. Verify the **GetRecordingConfigurationResponse** messages from the DUT. Verify that RecordingConfiguration.Source.Name, RecordingConfiguration.Source.SourceId, RecordingConfiguration.Source.Address settings were applied.
9. Restore DUT settings.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **SetRecordingConfigurationResponse** message.
- The DUT did not send a valid **GetRecordingConfigurationResponse** message.
- The DUT did not apply RecordingConfiguration.Source.Name, RecordingConfiguration.Source.SourceId, RecordingConfiguration.Source.Address settings.

Note: See [Annex A.8](#) for Recording Source Information Parameters Length limitations.

5.4.10 DYNAMIC RECORDINGS CONFIGURATION (MAXIMUM LENGTH OF RECORDING SOURCE INFORMATION)

Test Case ID: RECORDING-4-1-12

Specification Coverage: Recording Source Information & wsdl (ONVIF Recording Control Service Specification)

Feature Under Test: CreateRecordingConfiguration, GetRecordingConfiguration

WSDL Reference: recording.wsdl, event.wsdl

Test Purpose: To verify applying of recording source information (Name, Address, SourceId) with maximum length by the DUT during Recording creation.

Pre-Requirement: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command. Dynamic Recording functionality is supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.7](#) for possibility to create a new recording.
4. ONVIF Client will invoke **CreateRecordingRequest** message with RecordingConfiguration.Source.Address = [valid URI with length 128 characters], RecordingConfiguration.Source.Name = [string with length 20 characters, which contains only readable characters and begins with letter], RecordingConfiguration.Source.Location = "LocationDescription", RecordingConfiguration.Source.Description = "Source Description", RecordingConfiguration.Source.SourceId = [valid URI with length 128 characters], RecordingConfiguration.Content = "Create recording event test", RecordingConfiguration.MaximumRetentionTime = [acceptable MaximumRetentionTime] to create a new recording.
5. Verify the **CreateRecordingResponse** message from the DUT (RecordingToken = RecordingToken1).
6. ONVIF Client will invoke **GetRecordingConfigurationRequest** message (RecordingToken = RecordingToken1) to get Recording configuration.
7. Verify the **GetRecordingConfigurationResponse** messages from the DUT. Verify that RecordingConfiguration.Source.Name, RecordingConfiguration.Source.SourceId, RecordingConfiguration.Source.Address settings were applied.
8. Restore DUT settings.

Test Result:

PASS –

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **CreateRecordingResponse** message.
- The DUT did not send a valid **GetRecordingConfigurationResponse** message.
- The DUT did not apply RecordingConfiguration.Source.Name, RecordingConfiguration.Source.SourceId, RecordingConfiguration.Source.Address settings.

Note: See [Annex A.8](#) for Recording Source Information Parameters Length limitations.

5.4.11 GET RECORDING JOB CONFIGURATION WITH INVALID TOKEN

Test Case ID: RECORDING-4-1-13

Specification Coverage: GetRecordingJobConfiguration (ONVIF Recording Control Service Specification)

Feature Under Test: GetRecordingJobConfiguration

WSDL Reference: recording.wsdl

Test Purpose: To verify Get Recording Job Configuration with invalid Token.

Pre-Requirement: ONVIF Client gets the entry point for Recording Control Service by GetServices command. Media Service or Receiver Service was received from the DUT. At least one media profile compatible with at least one existing or created recording exists on the DUT. Options are supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingJobsRequest** message to retrieve a complete recording jobs list.
4. If there is at least one RecordingJob, then skip steps 5-16 and go to the step 11.
5. If Receiver service is supported, then ONVIF Client execute [Annex A.9](#) with RequiredSpareJobs = 0 to create or select Recording to make sure that 1 recording job can be created.
6. ONVIF Client will invoke **CreateRecordingJobRequest** message (JobConfiguration.RecordingToken = "RecordingToken1", JobConfiguration.Mode =

"Idle", JobConfiguration.Priority = 1, no JobConfiguration.Source.SourceToken.Token, JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Receiver", JobConfiguration.Source.AutoCreateReceiver = true) to create a recording job and auto create receiver.

7. Verify **CreateRecordingJobResponse** from the DUT and go to step 10.
8. If Receiver service is not supported ONVIF Client execute [Annex A.12](#) with RequiredSpareJobs = 0 to create or select Recording (RecordingToken = RecordingToken1) to make sure that 1 recording job can be created and to get the compatible media profile tokens list.
9. ONVIF Client will invoke **CreateRecordingJobRequest** message (JobConfiguration.RecordingToken = RecordingToken1, JobConfiguration.Mode = Idle, JobConfiguration.Priority = 1, JobConfiguration.Source.SourceToken.Token = "ProfileToken1", where ProfileToken1 is token of MediaProfile from Compatible Sources list, JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Profile", JobConfiguration.Source.AutoCreateReceiver is not present) to create a recording job with Idle mode.
10. Verify the **CreateRecordingJobResponse** message from the DUT.
11. ONVIF Client will invoke **GetRecordingJobConfigurationRequest** message (invalid JobToken).
12. The DUT will generate SOAP 1.2 fault message (InvalidArgVal/NoRecordingJob).
13. ONVIF Client restores recording jobs if they were changed in step 9.

Test Result:

PASS –

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetRecordingsResponse** message.
- The DUT did not send a valid **CreateRecordingJobResponse** message.
- The DUT did not send SOAP 1.2 fault message at step 11.
- The DUT sent an incorrect SOAP 1.2 fault message (fault code, namespace, etc.) at step 11.

Note: Other faults than specified in the test are acceptable, though the specified are preferable.

5.4.12 GET RECORDING JOB STATE WITH INVALID TOKEN

Test Case ID: RECORDING-4-1-14

Specification Coverage: GetRecordingJobState (ONVIF Recording Control Service Specification)

Feature Under Test: GetRecordingJobState

WSDL Reference: recording.wsdl

Test Purpose: To verify Get Recording Job State with invalid Token.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command. At least one media profile compatible with at least one existing or created recording exists on the DUT. Options are supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingJobsRequest** message to retrieve a complete recording jobs list.
4. If there is at least one RecordingJob, then skip steps 5-10 and go to the step 11.
5. If Receiver service is supported, then ONVIF Client execute [Annex A.9](#) with RequiredSpareJobs = 0 to create or select Recording to make sure that 1 recording job can be created.
6. ONVIF Client will invoke **CreateRecordingJobRequest** message (JobConfiguration.RecordingToken = "RecordingToken1", JobConfiguration.Mode = "Idle", JobConfiguration.Priority = 1, no JobConfiguration.Source.SourceToken.Token, JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Receiver", JobConfiguration.Source.AutoCreateReceiver = true) to create a recording job and auto create receiver.
7. Verify **CreateRecordingJobResponse** from the DUT and go to step 11.
8. If Receiver service is not supported ONVIF Client execute [Annex A.12](#) with RequiredSpareJobs = 0 to create or select Recording (RecordingToken = RecordingToken1) to make sure that 1 recording job can be created and to get the compatible media profile tokens list.

9. ONVIF Client will invoke **CreateRecordingJobRequest** message (JobConfiguration.RecordingToken = RecordingToken1, JobConfiguration.Mode = Idle, JobConfiguration.Priority = 1, JobConfiguration.Source.SourceToken.Token = "ProfileToken1", where ProfileToken1 is token of MediaProfile from Compatible Sources list, JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Profile", JobConfiguration.Source.AutoCreateReceiver is not present) to create a recording job with Idle mode.
10. Verify the **CreateRecordingJobResponse** message from the DUT.
11. ONVIF Client will invoke **GetRecordingJobStateRequest** message (invalid JobToken).
12. The DUT will generate SOAP 1.2 fault message (InvalidArgVal/NoRecordingJob).
13. ONVIF Client restores recording jobs if they were changed in step 9.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a valid **GetRecordingsResponse** message.
- The DUT did not send a valid **GetRecordingJobsResponse** message.
- The DUT did not send a valid **CreateRecordingJobResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message
- The DUT did not send SOAP 1.2 fault message.
- The DUT sent an incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

Note: Other faults than specified in the test are acceptable, though the specified are preferable.

5.5 Events

5.5.1 RECORDING CONTROL – RECORDING CONFIGURATION EVENT

Test Case ID: RECORDING-5-1-3

Specification Coverage: Configuration changes (ONVIF Recording Control Service Specification)

Feature Under Test: GetRecordingConfiguration, SetRecordingConfiguration

WSDL Reference: recording.wsdl, event.wsdl

Test Purpose: To verify tns1:RecordingConfig/RecordingConfiguration event generation after recording configuration change, to verify tns1:RecordingConfig/RecordingConfiguration event format.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command. At least one recording exists for a DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.
4. Verify the **GetRecordingsResponse** message from the DUT. Select one of the recordings to use it in the further steps.
5. ONVIF Client will invoke **GetEventPropertiesRequest** message to retrieve all events supported by the DUT.
6. Verify the **GetEventPropertiesResponse** message from the DUT.
7. Check if there is an event with Topic tns1:RecordingConfig/RecordingConfiguration. If there is no event with such Topic skip other steps, fail the test and go to the next test.
8. Check that this event is not a Property event (MessageDescription.IsProperty = false).
9. Check that this event contains Source.SimpleItemDescription item with Name = "RecordingToken" and Type = "tt:RecordingReference".
10. Check that this event contains Data.ElementItemDescription item with Name = "Configuration" and Type = "tt:RecordingConfiguration".
11. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/RecordingConfiguration Topic as Filter and an InitialTerminationTime of timeout1.
12. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.

13. ONVIF Client will invoke **SetRecordingConfigurationRequest** message (RecordingToken = [selected recording], RecordingConfiguration.Source.Name = [other than current], other Recording Configuration parameters without change) to configure Recording.
14. Verify the **SetRecordingConfigurationResponse** message from the DUT.
15. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
16. Verify that the DUT sends a **PullMessagesResponse** that contains NotificationMessages. Repeat step 15 until Notification for selected Recording is received.
17. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
18. Verify that TopicExpression is equal to tns1:RecordingConfig/RecordingConfiguration for received Notify message.
19. Verify that notification contains Source.SimpleItem item with Name = "RecordingToken" and Value is equal to selected Recording Token.
20. Verify that each notification contains Data.ElementItem item with Name = "Configuration" and Value with type is equal to tt:RecordingConfiguration (validation with XML Schema complex type).
21. ONVIF Client will invoke **GetRecordingConfigurationRequest** message for selected Recording.
22. Verify the **GetRecordingConfigurationResponse** messages from the DUT. Verify that Data.ElementItem item with Name = "Configuration" from Notification message has the same value with RecordingConfiguration element from **GetRecordingConfigurationResponse** messages for selected Recording.
23. Restore DUT settings.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a **GetEventPropertiesResponse** message.
- The DUT did not send **CreatePullPointSubscriptionResponse** message.

- The DUT did not send **PullMessagesResponse** message.
- The DUT did not send a valid SubscriptionReference.
- The DUT did not send a valid **SetRecordingConfigurationResponse** message.
- The DUT did not send a valid **GetRecordingConfigurationResponse** message.
- The DUT did not send a valid **GetRecordingsResponse** message with at least one recording.
- The DUT did not send a Notification message that contains an event tns1:RecordingConfig/RecordingConfiguration for selected Recording.
- The DUT sent an invalid Notification message (no corresponding Source.SimpleItem, Data.SimpleItem, or Data.ElementItem wrong type of Value fields, invalid RecordingToken or Configuration values).
- The DUT does not return valid tns1:RecordingConfig/RecordingConfiguration Topic in **GetEventPropertiesResponse**.

Note: The Subscription Manager has to be deleted at the end of the test either by calling unsubscribe or through a timeout.

Note: ONVIF Client at step 16 will wait for Notification message until notification for selected Recording is received or Operation Delay after last notification expires.

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires. If DUT returns UnacceptableTerminationTimeFault, resend Renew request with acceptable InitialTerminationTime from UnacceptableTerminationTimeFault.

Note: If DUT cannot accept the set value to Timeout or MessageLimit, ONVIF Client retries to send the PullMessage message with Timeout and MessageLimit which is contained in PullMessagesFaultResponse.

Note: timeout1 will be taken from Subscription Timeout field of ONVIF Device Test Tool.

Note: See [Annex A.8](#) for Recording Source Information Parameters Length limitations.

5.5.2 RECORDING CONTROL – TRACK CONFIGURATION EVENT

Test Case ID: RECORDING-5-1-4

Specification Coverage: Configuration changes (ONVIF Recording Control Service Specification)

Feature Under Test: GetTrackConfiguration, SetTrackConfiguration

WSDL Reference: recording.wsdl, event.wsdl

Test Purpose: To verify tns1:RecordingConfig/TrackConfiguration event generation after track configuration change, to verify tns1:RecordingConfig/TrackConfiguration event format.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command. At least one recording with track exists for the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.
4. Verify the **GetRecordingsResponse** message from the DUT. Select one of the recording tracks to use it in the further steps.
5. ONVIF Client will invoke **GetEventPropertiesRequest** message to retrieve all events supported by the DUT.
6. Verify the **GetEventPropertiesResponse** message from the DUT.
7. Check if there is an event with Topic tns1:RecordingConfig/TrackConfiguration. If there is no event with such Topic, skip other steps, fail the test and go to the next test.
8. Check that this event is not a Property event (MessageDescription.IsProperty = false).
9. Check that this event contains Source.SimpleItemDescription item with Name = "RecordingToken" and Type = "tt:RecordingReference".
10. Check that this event contains Source.SimpleItemDescription item with Name = "TrackToken" and Type = "tt:TrackReference".
11. Check that this event contains Data.ElementItemDescription item with Name = "Configuration" and Type = "tt:TrackConfiguration".
12. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/TrackConfiguration Topic as Filter and an InitialTerminationTime of timeout1.

13. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
14. ONVIF Client will invoke **SetTrackConfigurationRequest** message (RecordingToken = [selected recording], TrackToken = [selected track], TrackConfiguration.Description = [other than current], other Recording Configuration parameters without change) to configure Track.
15. Verify the **SetTrackConfigurationResponse** message from the DUT.
16. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
17. Verify that the DUT sends a **PullMessagesResponse** that contains NotificationMessages. Repeat step 16 until Notification for selected Track is received.
18. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
19. Verify that TopicExpression is equal to tns1:RecordingConfig/TrackConfiguration for received Notify message.
20. Verify that notification contains Source.SimpleItem item with Name = "RecordingToken" and Value is equal to selected Recording Token.
21. Verify that notification contains Source.SimpleItem item with Name = "TrackToken" and Value is equal to selected Track Token.
22. Verify that each notification contains Data.ElementItem item with Name = "Configuration" and Value with type is equal to tt:TrackConfiguration (validation with XML Schema complex type).
23. ONVIF Client will invoke **GetTrackConfigurationRequest** message for selected Track and corresponding Recording.
24. Verify the **GetTrackConfigurationResponse** messages from the DUT. Verify that Data.ElementItem item with Name = "Configuration" from Notification message has the same value with TrackConfiguration element from **GetTrackConfigurationResponse** messages for selected Track.
25. Restore DUT settings.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a **GetEventPropertiesResponse** message.
- The DUT did not send **CreatePullPointSubscriptionResponse** message.
- The DUT did not send **PullMessagesResponse** message.
- The DUT did not send a valid SubscriptionReference.
- The DUT did not send a valid **SetTrackConfigurationResponse** message.
- The DUT did not send a valid **GetTrackConfigurationResponse** message.
- The DUT did not send a valid **GetRecordingsResponse** message with at least one recording with track.
- The DUT did not send a Notification message that contains an event tns1:RecordingConfig/TrackConfiguration for selected Recording.
- The DUT sent an invalid Notification message (no corresponding Source.SimpleItem, Data.SimpleItem, or Data.ElementItem wrong type of Value fields, invalid RecordingToken, TrackToken, or Configuration values).
- The DUT does not return valid tns1:RecordingConfig/TrackConfiguration Topic in **GetEventPropertiesResponse**.

Note: The Subscription Manager has to be deleted at the end of the test either by calling unsubscribe or through a timeout.

Note: ONVIF Client at step 17 will wait for Notification message until notification for selected Track is received or Operation Delay after last notification expires.

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires. If DUT returns UnacceptableTerminationTimeFault, re-send Renew request with acceptable InitialTerminationTime from UnacceptableTerminationTimeFault.

Note: If DUT cannot accept the set value to Timeout or MessageLimit, ONVIF Client retries to send the PullMessage message with Timeout and MessageLimit which is contained in PullMessagesFaultResponse.

Note: timeout1 will be taken from Subscription Timeout field of ONVIF Device Test Tool.

5.5.3 RECORDING CONTROL – CREATE RECORDING EVENT

Test Case ID: RECORDING-5-1-6

Specification Coverage: Recording and track creation and deletion (ONVIF Recording Control Service Specification)

Feature Under Test: CreateRecordingConfiguration

WSDL Reference: recording.wsdl, event.wsdl

Test Purpose: To verify tns1:RecordingConfig/CreateRecording event generation after recording creation, to verify tns1:RecordingConfig/CreateRecording event format.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command. Dynamic Recording functionality is supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.7](#) for possibility to create a new recording.
4. ONVIF Client will invoke **GetEventPropertiesRequest** message to retrieve all events supported by the DUT.
5. Verify the **GetEventPropertiesResponse** message from the DUT.
6. Check if there is an event with Topic tns1:RecordingConfig/CreateRecording. If there is no event with such Topic, skip other steps, fail the test and go to the next test.
7. Check that this event is not a Property event (MessageDescription.IsProperty = false).
8. Check that this event contains Source.SimpleItemDescription item with Name = "RecordingToken" and Type = "tt:RecordingReference".
9. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/CreateRecording Topic as Filter and an InitialTerminationTime of timeout1.
10. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
11. ONVIF Client will invoke **CreateRecordingRequest** message with
RecordingConfiguration.Source.SourceId as any URI,
RecordingConfiguration.Source.Name = "CameraName",

RecordingConfiguration.Source.Location = "LocationDescription",
RecordingConfiguration.Source.Description = "Source Description",
RecordingConfiguration.Source.Address as address of the device,
RecordingConfiguration.Content = "Create recording event test",
RecordingConfiguration.MaximumRetentionTime = [acceptable MaximumRetentionTime] to
create a new recording.

12. Verify the **CreateRecordingResponse** message from the DUT (RecordingToken = RecordingToken1).
13. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
14. Verify that the DUT sends a **PullMessagesResponse** that contains NotificationMessages. Repeat step 13 until Notification for created Recording is received.
15. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
16. Verify that TopicExpression is equal to tns1:RecordingConfig/CreateRecording for received Notify message.
17. Verify that notification contains Source.SimpleItem item with Name = "RecordingToken" and Value is equal to created Recording Token.
18. Restore DUT settings.

Test Result:

PASS –

- The DUT passes all assertions.

FAIL –

- The DUT did not send a **GetEventPropertiesResponse** message.
- The DUT did not send **CreatePullPointSubscriptionResponse** message.
- The DUT did not send **PullMessagesResponse** message.
- The DUT did not send a valid SubscriptionReference.
- The DUT did not send a valid **CreateRecordingResponse** message.
- The DUT did not send a Notification message that contains an event tns1:RecordingConfig/CreateRecording for created Recording.

- The DUT sent an invalid Notification message (no corresponding Source.SimpleItem, Data.SimpleItem, wrong type of Value fields, invalid RecordingToken).
- The DUT does not return a valid tns1:RecordingConfig/CreateRecording Topic in **GetEventPropertiesResponse**.

Note: The Subscription Manager has to be deleted at the end of the test either by calling unsubscribe or through a timeout.

Note: ONVIF Client at step 14 will wait for Notification message until notification for created Recording is received or Operation Delay after last notification expires.

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires. If DUT returns UnacceptableTerminationTimeFault, resend Renew request with acceptable InitialTerminationTime from UnacceptableTerminationTimeFault.

Note: If DUT cannot accept the set value to Timeout or MessageLimit, ONVIF Client retries to send the PullMessage message with Timeout and MessageLimit which is contained in PullMessagesFaultResponse.

Note: timeout1 will be taken from Subscription Timeout field of ONVIF Device Test Tool.

Note: See [Annex A.8](#) for Recording Source Information Parameters Length limitations.

5.5.4 RECORDING CONTROL – CREATE TRACK EVENT

Test Case ID: RECORDING-5-1-8

Specification Coverage: Recording and track creation and deletion (ONVIF Recording Control Service Specification)

Feature Under Test: None

WSDL Reference: event.wsdl

Test Purpose: To verify tns1:RecordingConfig/CreateTrack event format in TopicSet.

Pre-Requisite: Event Service is supported by the DUT. Dynamic Recording or dynamic track functionality is supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.

3. ONVIF Client will invoke **GetEventPropertiesRequest** message to retrieve all events supported by the DUT.
4. Verify the **GetEventPropertiesResponse** message from the DUT.
5. Check if there is an event with Topic tns1:RecordingConfig/CreateTrack. If there is no event with such Topic, skip other steps, fail the test and go to the next test.
6. Check that this event is not a Property event (MessageDescription.IsProperty = false).
7. Check that this event contains Source.SimpleItemDescription item with Name = "RecordingToken" and Type = "tt:RecordingReference".
8. Check that this event contains Source.SimpleItemDescription item with Name = "TrackToken" and Type = "tt:TrackReference".

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a **GetEventPropertiesResponse** message.
- The DUT does not return valid tns1:RecordingConfig/CreateTrack Topic in **GetEventPropertiesResponse**.

5.5.5 RECORDING CONTROL – DELETE TRACK EVENT

Test Case ID: RECORDING-5-1-9

Specification Coverage: Recording and track creation and deletion (ONVIF Recording Control Service Specification)

Feature Under Test: None

WSDL Reference: event.wsdl

Test Purpose: To verify tns1:RecordingConfig/DeleteTrack event format in TopicSet.

Pre-Requisite: Event Service is supported by the DUT. Dynamic Recording or dynamic track functionality is supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetEventPropertiesRequest** message to retrieve all events supported by the DUT.
4. Verify the **GetEventPropertiesResponse** message from the DUT.
5. Check if there is an event with Topic tns1:RecordingConfig/DeleteTrack. If there is no event with such Topic skip other steps, fail the test and go to the next test.
6. Check that this event is not a Property event (MessageDescription.IsProperty = false).
7. Check that this event contains Source.SimpleItemDescription item with Name = "RecordingToken" and Type = "tt:RecordingReference".
8. Check that this event contains Source.SimpleItemDescription item with Name = "TrackToken" and Type = "tt:TrackReference".

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a **GetEventPropertiesResponse** message.
- The DUT does not return a valid tns1:RecordingConfig/DeleteTrack Topic in **GetEventPropertiesResponse**.

5.5.6 RECORDING CONTROL – CREATE TRACK EVENT (CREATE RECORDING)

Test Case ID: RECORDING-5-1-10

Specification Coverage: Recording and track creation and deletion (ONVIF Recording Control Service Specification)

Feature Under Test: CreateRecording

WSDL Reference: recording.wsdl, event.wsdl

Test Purpose: To verify tns1:RecordingConfig/CreateTrack event generation after recording creation, to verify tns1:RecordingConfig/CreateTrack event format.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command. Dynamic Recording functionality is supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.7](#) for possibility to create a new recording.
4. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/CreateTrack Topic as Filter and an InitialTerminationTime of timeout1.
5. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
6. ONVIF Client will invoke **CreateRecordingRequest** message with
RecordingConfiguration.Source.SourceId as any URI,
RecordingConfiguration.Source.Name = "CameraName",
RecordingConfiguration.Source.Location = "LocationDescription",
RecordingConfiguration.Source.Description = "Source Description",
RecordingConfiguration.Source.Address as address of the device,
RecordingConfiguration.Content = "Create recording event test",
RecordingConfiguration.MaximumRetentionTime = [acceptable MaximumRetentionTime] to create a new recording with default tracks.
7. Verify the **CreateRecordingResponse** message from the DUT (RecordingToken = RecordingToken1).
8. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 2.
9. Verify that the DUT sends a **PullMessagesResponse** that contains NotificationMessages. Repeat step 8 until Notifications for all default tracks for created Recording are received.
10. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
11. Verify that TopicExpression is equal to tns1:RecordingConfig/CreateTrack for all received Notify messages.

12. Verify that each notification contains Source.SimpleItem item with Name = "RecordingToken" and Value is equal to created Recording Token.
13. Verify that there is one notification for each default tracks of created recording (notification contains Source.SimpleItem item with Name = "TrackToken" with Value is equal to each Track Token).
14. Restore DUT settings.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send **CreatePullPointSubscriptionResponse** message.
- The DUT did not send **PullMessagesResponse** message.
- The DUT did not send a valid SubscriptionReference.
- The DUT did not send a valid **CreateRecordingResponse** message.
- The DUT did not send a Notification message that contains an event tns1:RecordingConfig/CreateTrack for at least one Track from created Recording.
- The DUT sent an invalid Notification message (no corresponding Source.SimpleItem, Data.SimpleItem, wrong type of Value fields, invalid TrackToken or RecordingToken).

Note: The Subscription Manager has to be deleted at the end of the test either by calling unsubscribe or through a timeout.

Note: ONVIF Client at step 9 will wait for Notification message until all notifications for each track for created Recording is received or Operation Delay after last notification expires.

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires. If DUT returns UnacceptableTerminationTimeFault, resend Renew request with acceptable InitialTerminationTime from UnacceptableTerminationTimeFault.

Note: If DUT cannot accept the set value to Timeout or MessageLimit, ONVIF Client retries to send the PullMessage message with Timeout and MessageLimit which is contained in PullMessagesFaultResponse.

Note: timeout1 will be taken from Subscription Timeout field of ONVIF Device Test Tool.

Note: See [Annex A.8](#) for Recording Source Information Parameters Length limitations.

5.5.7 RECORDING CONTROL – DELETE TRACK EVENT (DELETE RECORDING)

Test Case ID: RECORDING-5-1-11

Specification Coverage: Recording and track creation and deletion (ONVIF Recording Control Service Specification)

Feature Under Test: DeleteRecording

WSDL Reference: recording.wsdl, event.wsdl

Test Purpose: To verify tns1:RecordingConfig/DeleteTrack event generation after recording deletion, to verify tns1:RecordingConfig/DeleteTrack event format.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command. Dynamic Recording functionality is supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.7](#) for possibility to create a new recording.
4. ONVIF Client will invoke **CreateRecordingRequest** message with
RecordingConfiguration.Source.SourceId as any URI,
RecordingConfiguration.Source.Name = "CameraName",
RecordingConfiguration.Source.Location = "LocationDescription",
RecordingConfiguration.Source.Description = "Source Description",
RecordingConfiguration.Source.Address as address of the device,
RecordingConfiguration.Content = "Create recording event test",
RecordingConfiguration.MaximumRetentionTime = [acceptable MaximumRetentionTime] to
create a new recording with default tracks.
5. Verify the **CreateRecordingResponse** message from the DUT (RecordingToken = RecordingToken1).
6. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with
tns1:RecordingConfig/DeleteTrack Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.

8. ONVIF Client will invoke **DeleteRecordingRequest** message (RecordingToken = RecordingToken1) to delete the created recording with default tracks.
9. Verify the **DeleteRecordingResponse** message from the DUT.
10. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 2.
11. Verify that the DUT sends a **PullMessagesResponse** that contains NotificationMessages. Repeat step 10 until Notifications for all default tracks for created Recording are received.
12. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
13. Verify that TopicExpression is equal to tns1:RecordingConfig/DeleteTrack for all received Notify messages.
14. Verify that each notification contains Source.SimpleItem item with Name = "RecordingToken" and Value is equal to created Recording Token.
15. Verify that there is one notification for each default tracks of deleted recording (notification contains Source.SimpleItem item with Name = "TrackToken" with Value is equal to each Track Token).
16. Restore DUT settings.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send **CreatePullPointSubscriptionResponse** message.
- The DUT did not send **PullMessagesResponse** message.
- The DUT did not send a valid SubscriptionReference.
- The DUT did not send a valid **CreateRecordingResponse** message.
- The DUT did not send a Notification message that contains an event tns1:RecordingConfig/CreateTrack for at least one Track from created Recording.
- The DUT sent an invalid Notification message (no corresponding Source.SimpleItem, Data.SimpleItem, and wrong type of Value fields, invalid TrackToken or RecordingToken).

Note: The Subscription Manager has to be deleted at the end of the test either by calling unsubscribe or through a timeout.

Note: ONVIF Client at step 11 will wait for Notification message until all notifications for each track for deleted Recording is received or Operation Delay after last notification expires.

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires. If DUT returns UnacceptableTerminationTimeFault, resend Renew request with acceptable InitialTerminationTime from UnacceptableTerminationTimeFault.

Note: If DUT cannot accept the set value to Timeout or MessageLimit, ONVIF Client retries to send the PullMessage message with Timeout and MessageLimit which is contained in PullMessagesFaultResponse.

Note: timeout1 will be taken from Subscription Timeout field of ONVIF Device Test Tool.

Note: See [Annex A.8](#) for Recording Source Information Parameters Length limitations.

5.5.8 RECORDING CONTROL – DELETE TRACK DATA EVENT

Test Case ID: RECORDING-5-1-14

Specification Coverage: Data deletion (ONVIF Recording Control Service Specification)

Feature Under Test: None

WSDL Reference: event.wsdl

Test Purpose: To verify tns1:RecordingConfig/DeleteTrackData event format in TopicSet.

Pre-Requisite: Event Service was received from the DUT. Dynamic Recording or dynamic track functionality is supported by the DUT. tns1:RecordingConfig/DeleteTrackData event is supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetEventPropertiesRequest** message to retrieve all events supported by the DUT.

4. Verify the **GetEventPropertiesResponse** message from the DUT.
5. Check if there is an event with Topic `tns1:RecordingConfig/DeleteTrackData`. If there is no event with such Topic, skip other steps, fail the test and go to the next test.
6. Check that this event is not a Property event (`MessageDescription.IsProperty = false`).
7. Check that this event contains `Source.SimpleItemDescription` item with `Name = "RecordingToken"` and `Type = "tt:RecordingReference"`.
8. Check that this event contains `Source.SimpleItemDescription` item with `Name = "TrackToken"` and `Type = "tt:TrackReference"`.
9. Check that this event contains `Data.SimpleItemDescription` item with `Name = "StartTime"` and `Type = "xs:dateTime"`.
10. Check that this event contains `Data.SimpleItemDescription` item with `Name = "EndTime"` and `Type = "xs:dateTime"`.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a **GetEventPropertiesResponse** message.
- The DUT does not return a valid `tns1:RecordingConfig/DeleteTrackData` Topic in **GetEventPropertiesResponse**.

5.5.9 RECORDING CONTROL – CREATE TRACK EVENT (CREATE TRACK)

Test Case ID: RECORDING-5-1-15**Specification Coverage:** Recording and track creation and deletion (ONVIF Recording Control Service Specification)**Feature Under Test:** CreateTrack**WSDL Reference:** `recording.wsdl`, `event.wsdl`**Test Purpose:** To verify `tns1:RecordingConfig/CreateTrack` event generation after track creation, to verify `tns1:RecordingConfig/CreateTrack` event format.

Pre-Requirement: ONVIF Client gets the entry point for Recording Control Service by GetServices command. Dynamic Track functionality is supported by the DUT. Options are supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.11](#) to create or select Recording to make sure that track creation procedure will be possible.
4. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/CreateTrack Topic as Filter and an InitialTerminationTime of timeout1.
5. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
6. ONVIF Client will invoke **CreateTrackRequest** message (RecordingToken=RecordingToken1 where RecordingToken1 is RecordingToken from [Annex A.11](#), TrackConfiguration.TrackType as track type from [Annex A.11](#), TrackConfiguration.Description = 'New Track') to create a new track in the recording.
7. Verify the **CreateTrackResponse** message (TrackToken = TrackToken1) from the DUT.
8. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 2.
9. Verify that the DUT sends a **PullMessagesResponse** that contains NotificationMessages. Repeat step 8 until Notification for created Track is received.
10. Verify received Notify message (correct value for UTC time, TopicExpression and wsnt:Message).
11. Verify that TopicExpression is equal to tns1:RecordingConfig/CreateTrack for received Notify message.
12. Verify that notification contains Source.SimpleItem item with Name = "RecordingToken" and Value is equal to selected Recording Token.
13. Verify that notification contains Source.SimpleItem item with Name = "TrackToken" and Value is equal to created Track Token.
14. Restore DUT settings.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send **CreatePullPointSubscriptionResponse** message.
- The DUT did not send **PullMessagesResponse** message.
- The DUT did not send a valid SubscriptionReference.
- The DUT did not send a valid **CreateTrackResponse** message
- The DUT will not allow creating new Track for selected Recording
- The DUT did not send a Notification message that contains an event tns1:RecordingConfig/ CreateTrack for created Track.
- The DUT sent an invalid Notification message (no corresponding Source.SimpleItem, Data.SimpleItem, wrong type of Value fields, invalid TrackToken or RecordingToken).

Note: The Subscription Manager has to be deleted at the end of the test either by calling unsubscribe or through a timeout.

Note: ONVIF Client at step 9 will wait for Notification message until Notification for created Track is received or Operation Delay after last notification expires.

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires. If DUT returns UnacceptableTerminationTimeFault, resend Renew request with acceptable InitialTerminationTime from UnacceptableTerminationTimeFault.

Note: If DUT cannot accept the set value to Timeout or MessageLimit, ONVIF Client retries to send the PullMessage message with Timeout and MessageLimit which is contained in PullMessagesFaultResponse.

Note: timeout1 will be taken from Subscription Timeout field of ONVIF Device Test Tool.

Note: DUT shall provide Recording with Options.Track.SpareTotal>0 to avoid track's data deletion.

5.5.10 RECORDING CONTROL – DELETE TRACK EVENT (DELETE TRACK)

Test Case ID: RECORDING-5-1-16

Specification Coverage: Recording and track creation and deletion (ONVIF Recording Control Service Specification)

Feature Under Test: DeleteTrack

WSDL Reference: recording.wsdl, event.wsdl

Test Purpose: To verify tns1:RecordingConfig/DeleteTrack event generation after track creation, to verify tns1:RecordingConfig/DeleteTrack event format.

Pre-Requirement: ONVIF Client gets the entry point for Recording Control Service by GetServices command. Dynamic Track functionality is supported by the DUT. Options are supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Execute [Annex A.11](#) to create or select Recording to make sure that track creation procedure will be possible.
4. ONVIF Client will invoke **CreateTrackRequest** message (RecordingToken=RecordingToken1 where RecordingToken1 is RecordingToken from [Annex A.11](#), TrackConfiguration.TrackType as track type from [Annex A.11](#), TrackConfiguration.Description = 'New Track') to create a new track in the recording.
5. Verify the **CreateTrackResponse** message (TrackToken = TrackToken1) from the DUT.
6. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/DeleteTrack Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
8. ONVIF Client will invoke **DeleteTrackRequest** message (RecordingToken = RecordingToken1, TrackToken = TrackToken1).
9. Verify the **DeleteTrackResponse** message from the DUT. In case SOAP 1.2 fault message repeat steps 7-8 for the next track. If there is no other tracks to be deleted fail the test and skip other steps.
10. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 2.

11. Verify that the DUT sends a **PullMessagesResponse** that contains NotificationMessages.
Repeat step 10 until Notification for created Track is received.
12. Verify received Notify message (correct value for UTC time, TopicExpression and wsnt:Message).
13. Verify that TopicExpression is equal to tns1:RecordingConfig/DeleteTrack for received Notify message.
14. Verify that notification contains Source.SimpleItem item with Name = "RecordingToken" and Value is equal to selected Recording Token.
15. Verify that notification contains Source.SimpleItem item with Name = "TrackToken" and Value is equal to deleted Track Token.
16. Restore DUT settings.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send **CreatePullPointSubscriptionResponse** message.
- The DUT did not send **PullMessagesResponse** message.
- The DUT did not send valid SubscriptionReference.
- The DUT did not send valid **CreateTrackResponse** message or corresponding SOAP 1.2 fault if it is not possible.
- The DUT will not allow deleting the Track for selected Recording.
- The DUT did not send valid **DeleteTrackResponse** message or corresponding SOAP 1.2 fault if it is not possible.
- The DUT did not send a Notification message that contains an event tns1:RecordingConfig/DeleteTrack for created Track.
- The DUT sent an invalid Notification message (no corresponding Source.SimpleItem, Data.SimpleItem, wrong type of Value fields, invalid TrackToken or RecordingToken).

Note: The Subscription Manager has to be deleted at the end of the test either by calling unsubscribe or through a timeout.

Note: ONVIF Client at step 11 will wait for Notification message until notification for deleted track is received or Operation Delay after last notification expires.

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires. If DUT returns UnacceptableTerminationTimeFault, resend Renew request with acceptable InitialTerminationTime from UnacceptableTerminationTimeFault.

Note: If DUT cannot accept the set value to Timeout or MessageLimit, ONVIF Client retries to send the PullMessage message with Timeout and MessageLimit which is contained in PullMessagesFaultResponse.

Note: timeout1 will be taken from Subscription Timeout field of ONVIF Device Test Tool.

Note: DUT shall provide Recording with Options.Track.SpareTotal > 0 to avoid track's data deletion.

5.5.11 RECORDING CONTROL – DELETE RECORDING EVENT

Test Case ID: RECORDING-5-1-17

Specification Coverage: Recording and track creation and deletion (ONVIF Recording Control Service Specification)

Feature Under Test: DeleteRecordingConfiguration

WSDL Reference: recording.wsdl, event.wsdl

Test Purpose: To verify tns1:RecordingConfig/DeleteRecording event generation after recording deletion, to verify tns1:RecordingConfig/DeleteRecording event format.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices/GetCapabilities command. Dynamic Recording functionality is supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **CreateRecordingRequest** message with

RecordingConfiguration.Source.SourceId	as	any	URI,
RecordingConfiguration.Source.Name	=		"CameraName",
RecordingConfiguration.Source.Location	=		"LocationDescription",

RecordingConfiguration.Source.Description = "Source Description",
RecordingConfiguration.Source.Address as address of the device,
RecordingConfiguration.Content = "Recording from device",
RecordingConfiguration.MaximumRetentionTime = "PT0S" to create a new recording.

4. Verify the **CreateRecordingResponse** message or SOAP 1.2 fault message from the DUT. If DUT returns **CreateRecordingResponse** message go to 7 step and use created recording.
5. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.
6. Verify the **GetRecordingsResponse** message from the DUT. ONVIF Client selects the first recording from the **GetRecordingsResponse** message.
7. ONVIF Client will invoke **GetEventPropertiesRequest** message to retrieve all events supported by the DUT.
8. Verify the **GetEventPropertiesResponse** message from the DUT.
9. Check if there is an event with Topic tns1:RecordingConfig/DeleteRecording. If there is no event with such Topic, skip other steps, fail the test and go to the next test.
10. Check that this event is not a Property event (MessageDescription.IsProperty = false).
11. Check that this event contains Source.SimpleItemDescription item with Name = "RecordingToken" and Type = "tt:RecordingReference".
12. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/DeleteRecording Topic as Filter and an InitialTerminationTime of timeout1.
13. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
14. ONVIF Client will invoke **DeleteRecordingRequest** for selected Recording.
15. Verify the **DeleteRecordingResponse** message from the DUT.
16. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
17. Verify that the DUT sends a **PullMessagesResponse** that contains NotificationMessages. Repeat step 16 until Notification for created Recording is received.
18. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).

19. Verify that TopicExpression is equal to tns1:RecordingConfig/DeleteRecording for received Notify message.
20. Verify that notification contains Source.SimpleItem item with Name = "RecordingToken" and Value is equal to deleted Recording Token.
21. Restore DUT settings.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a **GetEventPropertiesResponse** message.
- The DUT did not send **CreatePullPointSubscriptionResponse** message.
- The DUT did not send **PullMessagesResponse** message.
- The DUT did not send a valid SubscriptionReference.
- The DUT did not send a valid **DeleteRecordingResponse** message.
- The DUT did not send a Notification message that contains an event tns1:RecordingConfig/DeleteRecording for deleted Recording.
- The DUT sent an invalid Notification message (no corresponding Source.SimpleItem, Data.SimpleItem, wrong type of Value fields, invalid RecordingToken).
- The DUT does not return valid tns1:RecordingConfig/DeleteRecording Topic in **GetEventPropertiesResponse**.
- The DUT sends empty recording list at the step 6.

Note: The Subscription Manager has to be deleted at the end of the test either by calling unsubscribe or through a timeout.

Note: ONVIF Client at step 17 will wait for Notification message until notification for created Recording is received or Operation Delay after last notification expires.

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires. If DUT returns UnacceptableTerminationTimeFault, resend Renew request with acceptable InitialTerminationTime from UnacceptableTerminationTimeFault.

Note: If DUT cannot accept the set value to Timeout or MessageLimit, ONVIF Client retries to send the PullMessage message with Timeout and MessageLimit which is contained in PullMessagesFaultResponse.

Note: timeout1 will be taken from Subscription Timeout field of ONVIF Device Test Tool.

5.5.12 RECORDING CONTROL – JOB STATE EVENT

Test Case ID: RECORDING-5-1-18

Specification Coverage: Recording job state changes (ONVIF Recording Control Service Specification), Properties (ONVIF Core Specification)

Feature Under Test: GetRecordingJobState

WSDL Reference: event.wsdl, recording.wsdl

Test Purpose: To verify tns1:RecordingConfig/JobState event generation after subscription and to verify tns1:RecordingConfig/JobState event format.

Pre-Requisite: Event Service was received from the DUT. ONVIF Client gets the entry point for Recording Control Service by GetServices. Media Service or Receiver Service was received from the DUT. At least one recording exists. All recording jobs were stopped. At least one media profile compatible with at least one existing or created recording exists on the DUT. Options are supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingJobsRequest** message to retrieve a complete recording jobs list.
4. If there is at least one RecordingJob, then skip steps 5-10 and go to the step 11.
5. If Receiver service is supported, then ONVIF Client execute [Annex A.9](#) with RequiredSpareJobs = 0 to create or select Recording to make sure that 1 recording job can be created.
6. ONVIF Client will invoke **CreateRecordingJobRequest** message (JobConfiguration.RecordingToken = "RecordingToken1", JobConfiguration.Mode =

- "Idle", JobConfiguration.Priority = 1, no JobConfiguration.Source.SourceToken.Token, JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Receiver", JobConfiguration.Source.AutoCreateReceiver = true) to create a recording job and auto create receiver.
7. Verify **CreateRecordingJobResponse** from the DUT and go to step 11.
 8. If Receiver service is not supported, ONVIF Client execute [Annex A.12](#) with RequiredSpareJobs = 0 to create or select Recording (RecordingToken = RecordingToken1) to make sure that 1 recording job can be created and to get the compatible media profile tokens list.
 9. ONVIF Client will invoke **CreateRecordingJobRequest** message (JobConfiguration.RecordingToken = RecordingToken1, JobConfiguration.Mode = Idle, JobConfiguration.Priority = 1, JobConfiguration.Source.SourceToken.Token = "ProfileToken1", where ProfileToken1 is token of MediaProfile from Compatible Sources list, JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Profile", JobConfiguration.Source.AutoCreateReceiver is not present) to create a recording job with Idle mode.
 10. Verify the **CreateRecordingJobResponse** message from the DUT.
 11. ONVIF Client will invoke **GetEventPropertiesRequest** message to retrieve all events supported by the DUT.
 12. Verify the **GetEventPropertiesResponse** message from the DUT.
 13. Check if there is an event with Topic tns1:RecordingConfig/JobState. If there is no event with such Topic skip other steps, fail the test and go to the next test.
 14. Check that this event is a Property event (MessageDescription.IsProperty = true).
 15. Check that this event contains Source.SimpleItemDescription item with Name = "RecordingJobToken" and Type = "tt:RecordingJobReference".
 16. Check that this event contains Data.SimpleItemDescription item with Name = "State" and Type = "xs:string".
 17. Check that this event contains Data.ElementItemDescription item with Name = "Information" and Type = "tt:RecordingJobStateInformation".
 18. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/JobState Topic as Filter and an InitialTerminationTime of timeout1.
 19. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.

20. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 2.
21. Verify that the DUT sends a **PullMessagesResponse** that contains NotificationMessages. Repeat step 20 until Notifications for all Jobs are received.
22. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).
23. Verify that TopicExpression is equal to tns1:RecordingConfig/JobState for all received Notify messages.
24. Verify that each notification contains Source.SimpleItem item with Name = "RecordingJobToken" and Value is equal to one of existing Recording Job Tokens (e.g. complete list of Recording Jobs contains Recording Job with the same token). Verify that there are Notification messages for each Recording Job.
25. Verify that each notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to xs:string.
26. Verify that each notification contains Data.ElementItem item with Name = "Information" and Value with type is equal to tt:RecordingJobStateInformation (validation with XML Schema complex type).
27. Verify that Notify PropertyOperation = "Initialized".
28. ONVIF Client will invoke **GetRecordingJobStateRequest** message for each Recording Job with corresponding tokens.
29. Verify the **GetRecordingJobStateResponse** messages from the DUT. Verify that Data.ElementItem item with Name = "Information" from Notification message has the same value with State element from corresponding **GetRecordingJobStateResponse** messages for each Recording Job. Verify that Data.SimpleItem item with Name = "State" from Notification message has the same value with State.State element from corresponding **GetRecordingJobStateResponse** messages for each Recording Job.
30. Restore DUT settings.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a **GetEventPropertiesResponse**
- The DUT did not send **CreatePullPointSubscriptionResponse** message.
- The DUT did not send **PullMessagesResponse** message.
- The DUT did not send a valid SubscriptionReference.
- The DUT did not send a valid **GetRecordingJobsResponse** message.
- The DUT did not send a valid **CreateRecordingJobResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message
- The DUT did not send a Notification message that contains a property event tns1:RecordingConfig/JobState at least for one Recording Job.
- The DUT sent an invalid Notification message (no corresponding Source.SimpleItem, Data.SimpleItem, or Data.ElementItem wrong type of Value fields, invalid RecordingJobToken, State, or Information values, PropertyOperation is not equal to "Initialized").
- The DUT does not return valid tns1:RecordingConfig/JobState Topic in **GetEventPropertiesResponse**.

Note: The Subscription Manager has to be deleted at the end of the test either by calling unsubscribe or through a timeout.

Note: ONVIF Client at step 20 will wait for Notification messages until notification for all Recording Job is received or Operation Delay after last notification expires.

Note: The Renew has to be used for renew subscription during the test, if InitialTerminationTime expires. If DUT returns UnacceptableTerminationTimeFault, resend Renew request with acceptable InitialTerminationTime from UnacceptableTerminationTimeFault.

Note: If DUT cannot accept the set value to Timeout or MessageLimit, ONVIF Client retries to send the PullMessage message with Timeout and MessageLimit which is contained in PullMessagesFaultResponse.

Note: timeout1 will be taken from Subscription Timeout field of ONVIF Device Test Tool.

5.5.13 RECORDING CONTROL – JOB STATE CHANGE EVENT

Test Case ID: RECORDING-5-1-19

Specification Coverage: Recording job state changes (ONVIF Recording Control Service Specification), Properties (ONVIF Core Specification)

Feature Under Test: GetRecordingJobState

WSDL Reference: event.wsdl, recording.wsdl

Test Purpose: To verify tns1:RecordingConfig/JobState event generation after property was changed and to verify tns1:RecordingConfig/JobState event format. Options are supported by the DUT.

Pre-Requisite: Event Service was received from the DUT. ONVIF Client gets the entry point for Recording Control Service by GetServices command. Media Service or Receiver Service was received from the DUT. At least one recording exists. All recording jobs were stopped. At least one media profile compatible with at least one existing or created recording exists on the DUT. Options are supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT.

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingJobsRequest** message to retrieve a complete recording jobs list.
4. If there is at least one RecordingJob, then skip steps 5-10 and go to the step 11.
5. If Receiver service is supported, then ONVIF Client execute [Annex A.9](#) with RequiredSpareJobs = 0 to create or select Recording to make sure that 1 recording job can be created.
6. ONVIF Client will invoke **CreateRecordingJobRequest** message (JobConfiguration.RecordingToken = "RecordingToken1", JobConfiguration.Mode = "Idle", JobConfiguration.Priority = 1, no JobConfiguration.Source.SourceToken.Token, JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Receiver", JobConfiguration.Source.AutoCreateReceiver = true) to create a recording job and auto create receiver.
7. Verify **CreateRecordingJobResponse** from the DUT and go to step 11.
8. If Receiver service is not supported ONVIF Client execute [Annex A.12](#) with RequiredSpareJobs = 0 to create or select Recording (RecordingToken = RecordingToken1)

to make sure that 1 recording job can be created and to get the compatible media profile tokens list.

9. ONVIF Client will invoke **CreateRecordingJobRequest** message (JobConfiguration.RecordingToken = RecordingToken1, JobConfiguration.Mode = Idle, JobConfiguration.Priority = 1, JobConfiguration.Source.SourceToken.Token = "ProfileToken1", where ProfileToken1 is token of MediaProfile from Compatible Sources list, JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Profile", JobConfiguration.Source.AutoCreateReceiver is not present) to create a recording job with Idle mode.
10. Verify the **CreateRecordingJobResponse** message from the DUT.
11. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/JobState Topic as Filter and an InitialTerminationTime of timeout1.
12. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
13. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 2.
14. Verify that the DUT sends a **PullMessagesResponse** that contains NotificationMessages. Repeat step 14 until Notification for selected Recording Job is received.
15. ONVIF Client will invoke **SetRecordingJobModeRequest** message (JobToken = [selected job token], Mode = [other than current]) to change Recording Job state.
16. Verify **SetRecordingJobModeResponse** message from the DUT.
17. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 2.
18. Verify that the DUT sends a **PullMessagesResponse** that contains NotificationMessages. Repeat step 17 until Notification for selected Recording Job is received.
19. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).
20. Verify that TopicExpression is equal to tns1:RecordingConfig/JobState for received Notification message.
21. Verify that notification contains Source.SimpleItem item with Name = "RecordingJobToken" and Value is equal to one of existing Recording Job Tokens (e.g. complete list of Recording Jobs contains Recording Job with the same token). Verify that there are Notification messages for selected Recording Job.

22. Verify that notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to xs:string.
23. Verify that notification contains Data.ElementItem item with Name = "Information" and Value with type is equal to tt:RecordingJobStateInformation (validation with XML Schema complex type).
24. Verify that Notify PropertyOperation = "Changed".
25. ONVIF Client will invoke **GetRecordingJobStateRequest** message for selected Recording Job with corresponding tokens.
26. Verify the **GetRecordingJobStateResponse** messages from the DUT. Verify that Data.ElementItem item with Name = "Information" from Notification message has the same value with State element from corresponding **GetRecordingJobStateResponse** messages for selected Recording Job. Verify that Data.ElementItem item with Name = "State" from Notification message has the same value with State.State element from corresponding **GetRecordingJobStateResponse** messages for selected Recording Job.
27. Restore DUT settings.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send **CreatePullPointSubscriptionResponse** message.
- The DUT did not send **PullMessagesResponse** message.
- The DUT did not send **SetRecordingJobModeResponse** message.
- The DUT did not send a valid SubscriptionReference.
- The DUT did not send a valid **GetRecordingJobsResponse** message.
- The DUT did not send a valid **CreateRecordingJobResponse** message.
- The DUT did not send a valid **GetRecordingJobStateResponse** message
- The DUT did not send a Notification message that contains a property event tns1:RecordingConfig/JobState for selected Recording Job.
- The DUT sent an invalid Notification message (no corresponding Source.SimpleItem, Data.SimpleItem, or Data.ElementItem wrong type of Value fields, invalid

RecordingJobToken, State or Information values, PropertyOperation is not equal to "Changed").

Note: The Subscription Manager has to be deleted at the end of the test either by calling unsubscribe or through a timeout.

Note: ONVIF Client at steps 15 and 18 will wait for Notification messages until notification for selected Recording Job is received or Operation Delay after last notification expires.

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires. If DUT returns UnacceptableTerminationTimeFault, resend Renew request with acceptable InitialTerminationTime from UnacceptableTerminationTimeFault.

Note: If DUT cannot accept the set value to Timeout or MessageLimit, ONVIF Client retries to send the PullMessage message with Timeout and MessageLimit which is contained in PullMessagesFaultResponse.

Note: timeout1 will be taken from Subscription Timeout field of ONVIF Device Test Tool.

Note: All Notification messages for Recording Job other than selected will be ignored during this test.

5.5.14 RECORDING CONTROL – RECORDING JOB CONFIGURATION EVENT

Test Case ID: RECORDING-5-1-20

Specification Coverage: Configuration changes (ONVIF Recording Control Service Specification)

Feature Under Test: GetRecordingJobConfiguration, SetGetRecordingJobConfiguration

WSDL Reference: recording.wsdl, event.wsdl

Test Purpose: To verify tns1:RecordingConfig/RecordingJobConfiguration event generation after recording job configuration change, to verify tns1:RecordingConfig/RecordingJobConfiguration event format.

Pre-Requisite: ONVIF Client gets the entry point for Recording Control Service by GetServices command. Media Service or Receiver Service was received from the DUT. At least one media profile compatible with at least one existing or created recording exists on the DUT. Options are supported by the DUT. Device supports Pull-Point Notification feature.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetRecordingJobsRequest** message to retrieve a complete recording jobs list.
4. If there is at least one RecordingJob, then skip steps 5-10 and go to the step 11.
5. If Receiver service is supported, then ONVIF Client execute [Annex A.9](#) with RequiredSpareJobs = 0 to create or select Recording to make sure that 1 recording job can be created.
6. ONVIF Client will invoke **CreateRecordingJobRequest** message (JobConfiguration.RecordingToken = "RecordingToken1", JobConfiguration.Mode = "Idle", JobConfiguration.Priority = 1, no JobConfiguration.Source.SourceToken.Token, JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Receiver", JobConfiguration.Source.AutoCreateReceiver = true) to create a recording job and auto create receiver.
7. Verify **CreateRecordingJobResponse** from the DUT and go to step 11.
8. If Receiver service is not supported ONVIF Client execute [Annex A.12](#) with RequiredSpareJobs = 0 to create or select Recording (RecordingToken = RecordingToken1) to make sure that 1 recording job can be created and to get the compatible media profile tokens list.
9. ONVIF Client will invoke **CreateRecordingJobRequest** message (JobConfiguration.RecordingToken = RecordingToken1, JobConfiguration.Mode = Idle, JobConfiguration.Priority = 1, JobConfiguration.Source.SourceToken.Token = "ProfileToken1", where ProfileToken1 is token of MediaProfile from Compatible Sources list, JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Profile", JobConfiguration.Source.AutoCreateReceiver is not present) to create a recording job with Idle mode.
10. Verify the **CreateRecordingJobResponse** message from the DUT.
11. ONVIF Client will invoke **GetEventPropertiesRequest** message to retrieve all events supported by the DUT.
12. Verify the **GetEventPropertiesResponse** message from the DUT.
13. Check if there is an event with Topic tns1:RecordingConfig/RecordingJobConfiguration. If there is no event with such Topic skip other steps, fail the test and go to the next test.
14. Check that this event is not a Property event (MessageDescription.IsProperty = false).

15. Check that this event contains Source.SimpleItemDescription item with Name = "RecordingJobToken" and Type = "tt:RecordingJobReference".
16. Check that this event contains Data.ElementItemDescription item with Name = "Configuration" and Type = "tt:RecordingJobConfiguration".
17. ONVIF Client will invoke **CreatePullPointSubscriptionRequest** message with tns1:RecordingConfig/RecordingJobConfiguration Topic as Filter and an InitialTerminationTime of timeout1.
18. Verify that the DUT sends a **CreatePullPointSubscriptionResponse** message.
19. ONVIF Client will invoke SetRecordingJobConfigurationRequest message (JobToken = "JobToken1", JobConfiguration.Mode = [other than current], other Recording Job Configuration parameters without change) to configure Recording Job.
20. Verify the **SetRecordingJobConfigurationResponse** message from the DUT.
21. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
22. Verify that the DUT sends a **PullMessagesResponse** that contains NotificationMessages. Repeat step 21 until Notification for selected RecordingJob is received.
23. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
24. Verify that TopicExpression is equal to tns1:RecordingConfig/RecordingJobConfiguration for received Notify message.
25. Verify that notification contains Source.SimpleItem item with Name = "RecordingJobToken" and Value is equal to selected Recording Job Token.
26. Verify that each notification contains Data.ElementItem item with Name = "Configuration" and Value with type is equal to tt:RecordingJobConfiguration (validation with XML Schema complex type).
27. ONVIF Client will invoke **GetRecordingJobConfigurationRequest** message for selected Recording Job.
28. Verify the **GetRecordingJobConfigurationResponse** messages from the DUT. Verify that Data.ElementItem item with Name = "Configuration" from Notification message has the same value with JobConfiguration element from **GetRecordingJobConfigurationResponse** messages for selected Recording Job.
29. Restore DUT settings.

Test Result:**PASS –**

- The DUT passes all assertions.

FAIL –

- The DUT did not send a **GetEventPropertiesResponse** message.
- The DUT did not send **CreatePullPointSubscriptionResponse** message.
- The DUT did not send **PullMessagesResponse** message.
- The DUT did not send a valid SubscriptionReference.
- The DUT did not send a valid **SetRecordingJobConfigurationResponse** message.
- The DUT did not send a valid **GetRecordingJobConfigurationResponse** message.
- The DUT did not send a valid **GetRecordingJobsResponse** message.
- The DUT did not send a valid **CreateRecordingJobResponse** message.
- The DUT did not send a Notification message that contains an event tns1:RecordingConfig/RecordingJobConfiguration for selected Recording Job.
- The DUT sent an invalid Notification message (no corresponding Source.SimpleItem, Data.SimpleItem, or Data.ElementItem wrong type of Value fields, invalid RecordingJobToken or Configuration values).
- The DUT does not return a valid tns1:RecordingConfig/RecordingJobConfiguration Topic in **GetEventPropertiesResponse**.

Note: The Subscription Manager has to be deleted at the end of the test either by calling unsubscribe or through a timeout.

Note: ONVIF Client at step 21 will wait for Notification message until notification for selected Recording Job is received or Operation Delay after last notification expires.

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires. If DUT returns UnacceptableTerminationTimeFault, resend Renew request with acceptable InitialTerminationTime from UnacceptableTerminationTimeFault.

Note: If DUT cannot accept the set value to Timeout or MessageLimit, ONVIF Client retries to send the PullMessage message with Timeout and MessageLimit which is contained in PullMessagesFaultResponse.

Note: timeout1 will be taken from Subscription Timeout field of ONVIF Device Test Tool.

Annex A Helper Procedures and Additional Notes

A.1 Comparison of parameter values for the same recording in GetRecordingsResponse message and in GetRecordingConfigurationResponse message

Compare RecordingItem.Configuration item from GetRecordings response and RecordingConfiguration item from GetRecordingConfiguration response. Parameter values will be assumed as the same, if they have the same values for:

- Source.SourceId
- Source.Name
- Source.Location
- Source.Description
- Source.Address
- Content
- MaximumRetentionTime

A.2 Comparison of parameter values for the same recording job in GetRecordingJobsResponse message and in GetRecordingJobConfigurationResponse message

Compare JobConfiguration items from GetRecordingJobs response and from GetRecordingJobConfiguration response. Parameter values will be assumed as the same, if they have the same values for:

- JobConfiguration.RecordingToken
- JobConfiguration.Mode
- JobConfiguration.Priority
- JobConfiguration.Source.SourceToken.Token
- JobConfiguration.Source.SourceToken.Type
- JobConfiguration.Source.Tracks.SourceTag

- JobConfiguration.Source.Tracks.Destination

A.3 Comparison of parameter values for the same recording job in GetRecordingJobsResponse message and in GetRecordingJobStateResponse message

For comparing parameter values for the same recording job item from **GetRecordingJobsResponse** message and **GetRecordingJobStateResponse** message the following steps will be done:

1. Verify that for each RecordingJobSource from **GetRecordingJobsResponse** there is a corresponding item with the same SourceToken in RecordingJobStateInformation from **GetRecordingJobStateResponse**.
2. Verify that for each RecordingJobStateSource from **GetRecordingJobStateResponse** there is a corresponding item with the same SourceToken in RecordingJobConfiguration from **GetRecordingJobsResponse**.
3. For each pair of corresponding RecordingJobSource and RecordingJobStateSource check that for any RecordingJobTrack with some SourceTag and Destination there is a corresponding RecordingJobStateTrack with the same SourceTag and Destination in RecordingJobStateSource.

A.4 Comparison of parameter values for the same track in GetTrackConfigurationResponse message and in GetRecordingsResponse message

Compare RecordingItem.Tracks.Track.TrackToken.Configuration item from GetRecordings response and TrackConfiguration item from GetRecordingConfiguration response. Parameter values will be assumed as the same, if they have the same values for TrackType and Description.

A.5 PullMessages algorithm for check Recording Job State changing

If the state field of the RecordingJobStateInformation structure changes, the device sends the correspond event. Algorithm of **PullMessages** sending for receiving event with the expected state:

Pre-requisite: at the T1 moment the DUT received answer to request which should cause Recording Job State changing (i.e. at the T1 moment Recording Job State at the device has definitely been changed)

1. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 1 to find NotificationMessage containing event with required JobState for used Recording.
2. Verify **PullMessagesResponse** message.
3. If no events are returned and CurrentTime of sending **PullMessages** is more than T1+delta, where delta is Operation delay time, skip other steps. If no events returned and CurrentTime of sending **PullMessages** is less or equal than T1+delta go to the step 1.
4. If event is returned, check UTC Time of the received event. If UTC Time is more than T1+delta, skip other steps.
5. If PropertyOperation is equal to "Changed" check event. Otherwise, go to the step 1.
6. Find event with Source.SimpleItem item with Name = "RecordingJobToken" and Value is equal to "JobToken1", Data.SimpleItem item with Name = "State" and Value is equal to required state ("Active", "PartiallyActive" or "Idle").
7. If event is not found, go to the step 1, otherwise, go to the next step.
8. In the found Message check Values of State.State and State.Sources.State of Data.ElementItemDescription item. Check that values are equal to the required state ("Active", "PartiallyActive" or "Idle").

Test will be assumed as failed in case:

- The DUT did not send a valid **PullMessagesResponse** message.
- The DUT did not send NotificationMessage with event PropertyOperation = "Changed".
- There was no event which has Source.SimpleItemDescription item with Name = "RecordingJobToken" and Value is equal to "JobToken1", Data.SimpleItemDescription item with Name = "State" and Value is equal to required state.
- State.State parameter value of Data.ElementItemDescription item in NotificationMessage is not equal to required state.
- State.Sources.State parameter value of Data.ElementItemDescription item in NotificationMessage is not equal to required state.

A.6 PullMessages algorithm for check Receiver State changing

If the receiver changes state, the device sends a corresponding event. Algorithm of **PullMessages** sending for receiving event with expected state.

1. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 1 to find NotificationMessage containing event with required ReceiverState for used Recording.
2. Verify **PullMessagesResponse** message.
3. If no events returned and CurrentTime of sending **PullMessages** is more than T1+delta, where delta is Operation delay time, skip other steps. If no events are returned and CurrentTime of sending **PullMessages** is less or equal to T1+delta, go to the step 1.
4. If event is returned, check UTC Time of the received event. If UTC Time is more than T1+delta, skip other steps.
5. Find event with Source.SimpleItemDescription Description item with Name = "ReceiverToken" and Value is equal to "ReceiverToken1", Data.SimpleItemDescription Description item with Name = "NewState" and Value is equal to required receiver state.
6. If event is not found, go to the step 1, otherwise, go to the next step.

Test will be assumed as failed in case:

- The DUT did not send a valid **PullMessagesResponse** message.
- The DUT did not send NotificationMessage with event which has Source.SimpleItemDescription Description item with Name = "ReceiverToken" and Value is equal to "ReceiverToken1", Data.SimpleItemDescription Description item with Name = "NewState" and Value is equal to required receiver state.

A.7 Creation of Recording prerequisite

Sometimes it is impossible to create a new Recording for the reason that maximum number of recordings supported by the device has been reached. ONVIF Client follows the following procedure of recording deleting:

1. ONVIF Client will invoke **GetServiceCapabilitiesRequest** message to retrieve Capabilities for the recording service.
2. Verify **GetServiceCapabilitiesResponse** message.
3. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.
4. Verify the **GetRecordingsResponse** message from the DUT.
5. If total number of recordings is less than MaxRecordings Capabilities element, skip other steps and run test. If MaxRecordings Capabilities is skipped, return to test procedure steps and try performing CreateRecording step. In error case perform 6-8 steps from this Annex.

6. ONVIF Client will invoke **DeleteRecordingRequest** message (RecordingToken = Token1, where Token1 is the first recording token from the **GetRecordingsResponse**) to delete recording.
7. Verify the **DeleteRecordingResponse** message or SOAP 1.2 fault message (Action/ CannotDelete or others) from the DUT. If **DeleteRecordingResponse** message received skip other steps and run test.
8. Repeat steps 6-7 for the next recording token. If there is no next Recording Token then skip the test.

A.8 Recording Source Information Parameters Maximum Length

There are the following limitations on maximum length of recording source parameters that shall be used during tests by ONVIF Device Test Tool to prevent faults from the DUT:

1. SourceId shall be less than or equal to 128 characters.
2. Name shall be less than or equal to 20 characters.
3. Address shall be less than or equal to 128 characters.

Note: these limitations will not be used, if ONVIF Device Test Tool reuses values receiving from the DUT.

A.9 Selection or Creation of Recording for recording job creation

ONVIF Client retrieves DynamicRecording capabilities from the DUT and follows the following procedure of Recording creation or selection for using recording for recording job creation:

If DynamicRecording is True, ONVIF Client follows the following procedure of Recording creation:

1. ONVIF Client will invoke **CreateRecordingRequest** message with

RecordingConfiguration.Source.SourceId	as	any	URI,
RecordingConfiguration.Source.Name	=	"CameraName",	
RecordingConfiguration.Source.Location	=	"LocationDescription",	
RecordingConfiguration.Source.Description	=	"Source Description",	
RecordingConfiguration.Source.Address	as	address	of the device,
RecordingConfiguration.Content	=	"Recording	from device",
RecordingConfiguration.MaximumRetentionTime	=	"PT0S"	to create a new recording.

2. Verify the **CreateRecordingResponse** message from the DUT (RecordingToken = "RecordingToken1"). If SOAP 1.2 fault message was returned, than skip other steps and follow the procedure for DynamicRecording=false case.
3. Onvif Client will invoke **GetRecordingOptionsRequest** message (RecordingToken="RecordingToken1") to get number of spare jobs that can be created for the recording.
4. Verify **GetRecordingOptionsResponse** message from the DUT. If Options.Job.Spare > RequiredSpareJobs, then return to test and use created Recording.
5. ONVIF Client will try deleting the RecordingJob via DeleteRecordingJob command (JobToken as Token of the first job received from the **GetRecordingJobsResponse** message) and repeat steps 3-5.
6. Repeare step 5 with the next Recording Job.

If DynamicRecording is False or recording creation procedure has been failed, ONVIF Client follows the following procedure of Recording selection:

1. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.
2. Verify the **GetRecordingsResponse** message from the DUT.
3. ONVIF Client will invoke **GetRecordingOptionsRequest** message (RecordingToken as Token of the first recording in the **GetRecordingsResponse** message which have track type compliant with Capabilities.Encoding value in **GetServiceCapabilitiesResponse** message) to get number of spare jobs that can be created for the recording.
4. If Options.Job.Spare > RequiredSpareJobs, then return to the test and use this Recording.
5. Repeat steps 3-4 for the next RecordingToken from the **GetRecordingsResponse** message.
6. If there is no other Recording, then ONVIF Client will try deleting the RecordingJob via DeleteRecordingJob command (JobToken as Token of the first job received from the **GetRecordingJobsResponse** message) and repeat steps3-5.
7. Repeat step 6 with the next Recording Job.

Note: Test will be assumed as failed in case:

- The DUT did not send a valid **GetRecordingsResponse** message.
- The DUT did not send a valid **GetRecordingOptionsResponse** message.

Note: See [Annex A.8](#) for Recording Source Information Parameters Length limitations.

A.10 Auto Creation of Receiver

For recording data from receiver ONVIF Client follows the following procedure of Auto Receiver creation by Create Recording Job with Idle Mode and Auto Create Receiver parameter:

1. ONVIF Client will invoke **CreateRecordingJobRequest** message (JobConfiguration.RecordingToken = "RecordingToken1", JobConfiguration.Mode = "Idle", JobConfiguration.Priority = 1, no JobConfiguration.Source.SourceToken.Token, JobConfiguration.Source.SourceToken.Type = "http://www.onvif.org/ver10/schema/Receiver", JobConfiguration.Source.AutoCreateReceiver = true) to create a recording job and auto create receiver.
2. Verify the **CreateRecordingJobResponse** message (JobToken = "JobToken1").
3. Verify that the JobConfiguration returned from CreateRecordingJob is identical to the JobConfiguration passed into CreateRecordingJob, except for the ReceiverToken and the AutoCreateReceiver.
4. Check that **CreateRecordingJobResponse** message contains JobConfiguration.Source.SourceToken.Token = ReceiverToken1 with assigned receiver Token. Check that AutoCreateReceiver field is omitted.
5. ONVIF Client will invoke **GetReceiversRequest** message to retrieve full list of receivers.
6. Verify **GetReceiversResponse** message from the DUT. Check that list of all receivers contains receiver with token equal to ReceiverToken1.

Note: If Receiver was not created for the reason Maximum Number of receivers has been reached, then delete receiver manually and run Annex's steps again.

Note: Test will be assumed as failed in case:

- The DUT sent **CreateRecordingJobResponse** message with at least one of the following JobConfiguration parameters values of which differ from the ones sent in **CreateRecordingJobRequest** message: JobConfiguration.RecordingToken, JobConfiguration.Mode, and JobConfiguration.Priority.
- The DUT sent **CreateRecordingJobResponse** message which didn't contain JobConfiguration.Source.SourceToken.Token element with assigned receiver Token
- The DUT sent JobConfiguration.Source.SourceToken.Type in **CreateRecordingJobRequest** message differs from http://www.onvif.org/ver10/schema/Receiver.

- The DUT sent **CreateRecordingJobResponse** message which contained AutoCreateReceiver field.
- **GetReceiversResponse** message did not contain Recording with Token = JobConfiguration.Source.SourceToken.Token from **CreateRecordingJobResponse** message.

A.11 Selection of Recording for track creation

If during a test, creation of any track is required, ONVIF Client will follow the following procedure for selection of recording:

1. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.
2. Verify the **GetRecordingsResponse** message from the DUT.
3. ONVIF Client will invoke **GetRecordingOptionsRequest** message (RecordingToken as Token of the first recording in the **GetRecordingsResponse** message) to get total spare number of tracks that can be added to this recording.
4. If Options.Track.SpareTotal>0, then return to the test and use this Recording for creation of track. Type of creation track shall correspond to value of SpareVideo, SpareAudio or SpareMetadata greater than 0.
5. Repeat steps 3-4 for the next RecordingToken from the **GetRecordingsResponse** message.
6. If there are no the next Recording then try do delete Track from recording with track and repeat 4-5 steps.

Note: If list of recording items is empty, Onvif Client will retrieve DynamicRecording capability from the DUT. In case Dynamic capability is true Onvif Client will create recording via CreateRecording command. Otherwise Onvif Client fails the test.

Note: If DUT returns SOAP fault (Action/CannotDelete) message to DeleteTrack request, then ONVIF will try deleting other tracks.

A.12 Selection or Creation of Recording for recording job creation on a Media profile

ONVIF Client retrieves DynamicRecording capabilities from the DUT and follows the following procedure of Recoding creation or selection for using recording for recording job creation on a Media profile:

If DynamicRecording is True, ONVIF Client follows the following procedure of Recording creation:

1. ONVIF Client will invoke **CreateRecordingRequest** message with RecordingConfiguration.Source.SourceId as any URI, RecordingConfiguration.Source.Name = "CameraName", RecordingConfiguration.Source.Location = "LocationDescription", RecordingConfiguration.Source.Description = "Source Description", RecordingConfiguration.Source.Address as address of the device, RecordingConfiguration.Content = "Recording from device", RecordingConfiguration.MaximumRetentionTime = "PT0S" to create a new recording.
2. Verify the **CreateRecordingResponse** message from the DUT (RecordingToken = "RecordingToken1"). If SOAP 1.2 fault message was returned, then skip other steps and follow the procedure for DynamicRecording=false case.
3. ONVIF Client will invoke **GetRecordingOptionsRequest** message (RecordingToken="RecordingToken1") to get Compatible Sources list and number of spare jobs that can be created for the recording.
4. Verify **GetRecordingOptionsResponse** message from the DUT.
5. If Options.Job.CompatibleSources list is empty, then ONVIF Client skips other steps and marks the test as failed.
6. If Options.Job.Spare > RequiredSpareJobs return to the test and use created Recording.
7. Client will try deleting the RecordingJob via DeleteRecordingJob command (JobToken as Token of the first job received from the **GetRecordingJobsResponse** message).
8. Repeat 3-7 steps.
9. Delete the next recording job via DeleteRecordingJob command and repeat step 8.

If DynamicRecording is False or recording creation procedure has been failed, ONVIF Client follows the following procedure of Recording selection:

1. ONVIF Client will invoke **GetRecordingsRequest** message to retrieve a complete recordings list.
2. Verify the **GetRecordingsResponse** message from the DUT.
3. ONVIF Client will invoke **GetRecordingOptionsRequest** message (RecordingToken as Token of the first recording in the **GetRecordingsResponse** message) to get Compatible Sources list.
4. Verify **GetRecordingOptionsResponse** message from the DUT.

5. If Options.Job.CompatibleSources list is empty, then repeat 3-4 steps for the next recording. If there is no other recording, then ONVIF Client skips other steps and marks the test as failed.
6. If Options.Job.Spare > RequiredSpareJobs return to the test and use this Recording.
7. Repeat steps 3-6 for the next RecordingToken from the **GetRecordingsResponse** message.
8. If there is no other Recording then ONVIF Client will try to delete the RecordingJob via DeleteRecordingJob command (JobToken as Token of the first job received from the **GetRecordingJobsResponse** message).
9. Repeat 3-7 steps.
10. Delete the next recording job via DeleteRecordingJob command and repeat step 9.

Note: Test will be assumed as failed in case:

- The DUT did not send a valid **GetRecordingsResponse** message.
- The DUT did not send a valid **GetRecordingOptionsResponse** message.

Note: If new Recording was not created and there was no Recording with not empty Compatible Sources list then configure media profile or recording for test manually.

Note: See [Annex A.8](#) for Recording Source Information Parameters Length limitations.

A.13 PullMessages algorithm for check Recording Job State initializing

If Recording Job is created on the device, the device sends the corresponding event with "Initialized" PropertyOperation. Algorithm of **PullMessages** sending for receiving event with the expected state is the following:

1. ONVIF Client will invoke **PullMessages** command with a PullMessagesTimeout of 20s and a MessageLimit of 1 to find NotificationMessage containing event with required JobState for used Recording.
2. Verify **PullMessagesResponse** from the DUT.
3. Repeat steps 1-2 until Notification for JobToken = JobToken1 with Data.SimpleItem item with Name = "State" and Value is equal to "Active" or "PartiallyActive" is received or operation delay time has expired.

Note: All Notification messages except messages with the expected recording job state will be ignored.