Standardizing IP Connectivity for Physical Security

ONVIF®

Imaging Device Test Specification

Version 20.06

June 2020

REVISION HISTORY

Vers.	Date	Description
11.12	Dec 22, 2011	First issue
12.06	Jun 18, 2012	Imaging Service Capabilities test cases have been added.
13.06	Jun, 2013	The following test cases was added: IMAGING COMMAND SETIMAGINGSETTINGS
13.12	Dec, 2013	Minor changes
14.06	Jun, 2014	IMAGING COMMAND SETIMAGINGSETTINGS test case was changed
14.12	Dec, 2014	IMAGING COMMAND SETIMAGINGSETTINGS test case was changed
16.06	May, 2016	4.1.8 IMAGING COMMAND SETIMAGINGSETTINGS ADDITIONAL FEATURES has been added.
16.07	Jul 5, 2016	Removed Exposure.Window testing, several comments from Sony have been implemented
16.07	Jul 13, 2016	4.1.8 - Test sequence updated
16.07	Jul 26, 2016	Comments implemented
16.07	Aug 8, 2016	Comments from Hiroyuki Sano
16.09	September, 2016	IMAGING COMMAND SETIMAGINGSETTINGS ADDITIONAL FEATURES test case was changed Annex A.1 was changed.
17.01	Dec, 2016	Test Sequence updated for IMAGING COMMAND SETIMAGINGSETTINGS ADDITIONAL FEATURES test case.
17.06	Mar 3, 2017	The following test cases were added according to #1352: REALTIME PULLPOINT SUBSCRIPTION – IMAGE TOO BLURRY REALTIME PULLPOINT SUBSCRIPTION – IMAGE TOO DARK REALTIME PULLPOINT SUBSCRIPTION – IMAGE TOO BRIGHT REALTIME PULLPOINT SUBSCRIPTION – GLOBAL SCENE CHANGE
17.06	Mar 22, 2017	The following test case was added according to #1407: REALTIME PULLPOINT SUBSCRIPTION – MOTION ALARM
17.06	May 25, 2017	The following test cases were updated according to #1352: REALTIME PULLPOINT SUBSCRIPTION – IMAGE TOO BLURRY REALTIME PULLPOINT SUBSCRIPTION – IMAGE TOO DARK REALTIME PULLPOINT SUBSCRIPTION – IMAGE TOO BRIGHT REALTIME PULLPOINT SUBSCRIPTION – GLOBAL SCENE CHANGE

17.12	Sep 21, 2017	The following test cases were added according to #1482: GET IMAGING SETTINGS AND GET OPTIONS CONSISTENCY
17.12	Nov 21, 2017	The following annex was updated according to #1492: Annex A.1 Get Video Sources
18.06	Jun 21, 2018	Reformatting document using new template
18.12	Dec 21, 2018	Switching Hub description in 'Network Configuration for DUT' section was updated according to #1737
19.12	Jul 30, 2019	The following test case was added according to #1896: IMAGING-1-1-8 IMAGING COMMAND SETIMAGINGSETTINGS – INVALID SETTINGS (Note added)
19.12	Jul 30, 2019	The following test case was added according to #1642: IMAGING-1-1-3 IMAGING COMMAND GETOPTIONS (steps 5-7 added)
19.12	Oct 30, 2019	The following test case was added according to #1933: IMAGING-1-1-3 IMAGING COMMAND GETOPTIONS (step 8 added) IMAGING-1-1-15 IMAGING COMMAND SETIMAGINGSETTINGS ADDITIONAL FEATURES (step 62 added) IMAGING-1-1-16 GET IMAGING SETTINGS AND GET OPTIONS CONSISTENCY (step 4.10.5 added)
20.06	May 13, 2020	Pre-Requisite of the following test cases updated with adding of Pull-Point Notification feature according to #1999: IMAGING-4-1-1 REALTIME PULLPOINT SUBSCRIPTION – IMAGE TOO BLURRY IMAGING-4-1-2 REALTIME PULLPOINT SUBSCRIPTION – IMAGE TOO DARK IMAGING-4-1-3 REALTIME PULLPOINT SUBSCRIPTION – IMAGE TOO BRIGHT IMAGING-4-1-4 REALTIME PULLPOINT SUBSCRIPTION – GLOBAL SCENE CHANGE IMAGING-4-1-5 REALTIME PULLPOINT SUBSCRIPTION – MOTION ALARM

Table of Contents

1 Introduction

The goal of the ONVIF test specification set is to make it possible to realize fully interoperable IP physical security implementation from different vendors. The set of ONVIF test specification describes the test cases necessary to verify the [ONVIF Network Interface Specs] and [ONVIF Conformance] requirements4 and the test cases are to be basic inputs for some Profile specification requirements. It also describes the test framework, test setup, pre-requisites, test policies needed for the execution of the described test cases.

This ONVIF Imaging Test Specification acts as a supplementary document to the [ONVIF Network Interface Specs], illustrating test cases required 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 Imaging 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 Imaging service which is defined in [ONVIF Network Interface Specs].

The principal intended purposes are:

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

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

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

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

1.1.1 Imaging Setting

Imaging Setting test cases are covered for verification to get and set imaging parameters for BacklightCompensation, Brightness, ColorSaturation, Sharpness, Contrast, Exposure, Focus, Ir cut filter, WhiteBalance, WideDynamicRange, Stabilization, IR cur filter auto adjustment, ToneCompensation, Defogging, and NoiseReduction. It means that ***GetImagingSettings, GetOptions and SetImagingSettings*** commands are covered by this test case.

1.1.2 Focus Move

Focus Move test cases are covered for verification of capabilities to move the focus lens. ***GetMoveOptions, Move, Stop and GetStatus*** commands are covered by this test case.

1.1.3 Events

Events test cases cover verification of property events provided by Imaging Service. It means that the following events are covered by these test cases:

- tns1:VideoSource/ImageTooBlurry/AnalyticsService;
- tns1:VideoSource/ImageTooBlurry/ImagingService;
- tns1:VideoSource/ImageTooBlurry/RecordingService;
- tns1:VideoSource/ImageTooDark/AnalyticsService;
- tns1:VideoSource/ImageTooDark/ImagingService;
- tns1:VideoSource/ImageTooDark/RecordingService;
- tns1:VideoSource/ImageTooBright/AnalyticsService;
- tns1:VideoSource/ImageTooBright/ImagingService;
- tns1:VideoSource/ImageTooBright/RecordingService;
- tns1:VideoSource/GlobalSceneChange/AnalyticsService;
- tns1:VideoSource/GlobalSceneChange/ImagingService;
- tns1:VideoSource/GlobalSceneChange/RecordingService;

- tns1:VideoSource/MotionAlarm.

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 Imaging Specs] ONVIF Core Specifications:

  https://www.onvif.org/profiles/specifications/

- [ONVIF Base Test] ONVIF Base Device Test Specifications:

  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/

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 Imaging Service and tests. For the list of applicable general terms and definitions, please see [ONVIF Base Test].

BacklightCompensation	Backlight Compensation
Brightness	Image brightness
ColorSaturation	Color saturation of the image
Contrast	Contrast of the image
Defogging	Clarify image contrast effect
Exposure	Exposure
Focus	Focus
Imaging Service	Services for exposure time, gain and white balance parameters among others.
Ir cut filter	Infrared cutoff filter
IR cut filter auto adjust	Alternate IR cut filter depend on the brightness automatically.
Sharpness	Sharpness of the video image
Stabilization	Stabilize images against device vibration
Tone compensation	Image contrast compensation appropriately
WhiteBalance	White balance
WideDynamicRange	Wide dynamic range

4 Test Overview

This section describes the test setup procedure and prerequisites needed, and the test policies that should be followed for test case execution.

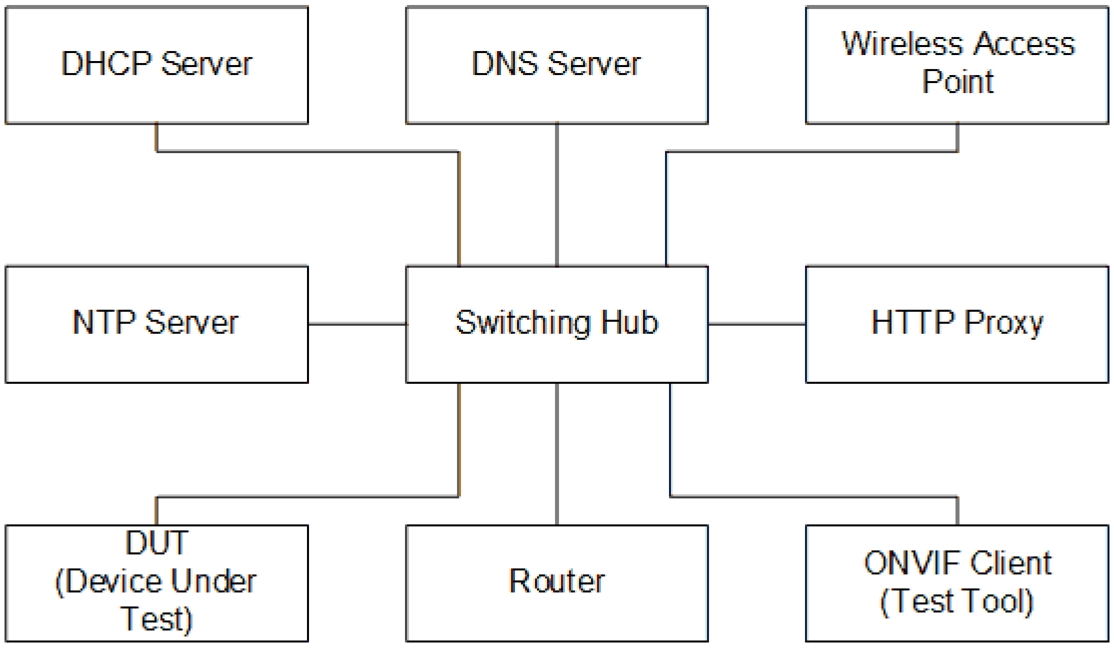
4.1 Test Setup

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 ONVIF Client Test Tool.
4. The DUT shall be configured with the time i.e. manual configuration of UTC time and if NTP is supported by DUT, then NTP time shall be synchronized with NTP Server.
5. The DUT time and ONVIF Client Test tool time shall be synchronized with each other either manually or by 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 Imaging Setting

DUT needs to support either Media service or Device IO service because of obtaining Video Source.

DUT shall give the Imaging Service entry point by GetCapabilities command. And Media service or Device IO service entry point also shall be provided by GetCapabilities command.

DUT shall support at least one Video Source and return the configuration through GetVideoSources command of Media service or Device IO.

4.3.2 Focus Move

Requirement is same with 4.3.1 Imaging Setting. If DUT supports remote focus control, DUT shall indicate it by returning GetMoveOptionsResponse with at least one parameter.

4.3.3 Events

DUT shall support GetEventProperties command and return all supported events in TopicSet.

DUT shall support Pull Point Subscription and Topic Expression filter.

DUT shall generate property events with initial state after subscription was done.

5 Imaging Test Cases

5.1 Imaging Settings

5.1.1 IMAGING COMMAND GETIMAGINGSETTINGS

Test Case ID: IMAGING-1-1-1

Specification Coverage: Get imaging settings

Feature under test: GetImagingSettings

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behaviour of GetImagingSettings command.

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 **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings for Video Source.
4. Verify the **GetImagingSettingsResponse** message from the DUT.
5. ONVIF Client will repeat steps 3-4 for other video sources supported by the DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- DUT did not send **GetImagingSettingsResponse** message.

5.1.2 IMAGING COMMAND GETOPTIONS

Test Case ID: IMAGING-1-1-3

Specification Coverage: Get options

Feature under test: GetOptions

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of GetOptions command.

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 **GetOptions** request (VideoSourceToken) to retrieve current imaging options for Video Source.
4. The DUT responds with **GetOptionsResponse** message with parameters
   - ImagingOptions =: *imagingOptions*
5. If *imagingOptions* contains Exposure.MinIris

   5.1. If *imagingOptions*.Exposure.MinIris.Min < 0, PASS step with warning.
6. If *imagingOptions* contains Exposure.MaxIris

   6.1. If *imagingOptions*.Exposure.MaxIris.Min < 0, PASS step with warning.
7. If *imagingOptions* contains Exposure.Iris

   7.1. If *imagingOptions*.Exposure.Iris.Min < 0, PASS step with warning.
8. If *imagingOptions* contains Focus.Extension.@AFModes
   - If *imagingOptions*.Focus.Extension.@AFModes list contains values differ from "OnceAfterMove", FAIL the test.
   - If *imagingOptions*.Focus does not contain at least one AutoFocusModes = "AUTO" , FAIL the test.
9. ONVIF Client will repeat steps 3-4 for other video sources supported by the DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetOptionsResponse** message.
- The DUT did not send a valid **GetOptionsResponse** message.
- The DUT returned **GetOptionsResponse** message with Min value greater than Max for at least one setting defined by Min and Max boundary:
    - BacklightCompensation.Level
    - Brightness
    - ColorSaturation
    - Contrast
    - Exposure.MinExposureTime
    - Exposure.MaxExposureTime
    - Exposure.MinGain
    - Exposure.MaxGain
    - Exposure.MinIris
    - Exposure.MaxIris
    - Exposure.ExposureTime
    - Exposure.Gain
    - Exposure.Iris
    - Focus.DefaultSpeed
    - Focus.NearLimit
    - Focus.FarLimit
    - Sharpness
    - WideDynamicRange.Level

    - WhiteBalance.YrGain
    - WhiteBalance.YbGain

- The DUT returned **GetOptionsResponse** message with the list of Exposure options, which does not correspond the list of possible Exposure.Mode:
    - If only Exposure.Mode=Manual supported, then there shall be no:
        - Exposure.Priority.Window
        - Exposure.MinExposureTime
        - Exposure.MaxExposureTime
        - Exposure.MinGain
        - Exposure.MaxGain
        - Exposure.MinIris
        - Exposure.MaxIris
    - If only Exposure.Mode=Auto supported, then there shall be no Exposure.ExposureTime, Exposure.Gain, Exposure.Iris.
- The DUT returned **GetOptionsResponse** message with BacklightCompensation.Level option if only BacklightCompensation.Mode=OFF.
- The DUT returned **GetOptionsResponse** message with the list of Focus options that does not correspond the list of possible Focus.Mode:
    - If only Focus.Mode=Manual supported, then there shall be no Focus.NearLimit, Focus.FarLimit.
    - If only Focus.Mode=Auto supported, then there shall be no Focus.DefaultSpeed.
- The DUT returned **GetOptionsResponse** message with the list of WhiteBalance options that does not correspond the list of possible WhiteBalance.Mode:
    - If only WhiteBalance.Mode=Auto supported, then there shall be no WhiteBalance.YrGain, WhiteBalance.YbGain.
- The DUT returned **GetOptionsResponse** message with the list of WideDynamicRange options that does not correspond the list of possible WideDynamicRange.Mode:

- If only WideDynamicRange.Mode=OFF supported, then there shall be no WideDynamicRange.Level.

Note: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition.

5.1.3 IMAGING COMMAND SETIMAGINGSETTINGS – INVALID SETTINGS

Test Case ID: IMAGING-1-1-8

Specification Coverage: Set imaging settings

Feature under test: SetImagingSettings

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of SetImagingSettings command in case of invalid ImagingSettings.

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 **GetOptions** request (VideoSourceToken) to retrieve current imaging options for Video Source.
4. Verify the **GetOptionsResponse** message from the DUT.
5. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.
6. Verify the **GetImagingSettingsResponse** message from the DUT.
7. ONVIF Client will invoke SetImagingSettingsRequest message (VideoSourceToken, all supported parameters from GetOptionsResponse set to invalid value, force persistence = false).
8. The DUT returns **env:Sender/ter:InvalidArgVal/ter:SettingsInvalid** SOAP 1.2 fault.

9. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

10. Verify the **GetImagingSettingsResponse** message from the DUT.

11. Verify that ImagingSettings is the same as **GetImagingSettingsResponse** message at step 6.

12. ONVIF Client will repeat steps 3-11 for other video sources supported by the DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetOptionsResponse** message.
- The DUT did not send a valid **GetOptionsResponse** message.
- The DUT did not send **GetImagingSettingsResponse** message.
- The DUT did not send a valid **GetImagingSettingsResponse** message.
- The DUT did not send SOAP 1.2 fault message for **SetImagingSettings** request.
- The DUT sent an incorrect SOAP 1.2 fault message (fault code, namespace, etc.).
- The DUT modified imaging settings by invoked **SetImagingSettings** request.

Note: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition.

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

Note: Steps 7-11 will be skipped if not possible to set invalid values.

5.1.4 IMAGING COMMAND GETIMAGINGSETTINGS – INVALID VIDEOSOURCETOKEN

Test Case ID: IMAGING-1-1-10

Specification Coverage: Get imaging settings

Feature under test: GetImagingSettings

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behaviour of GetImagingSettings command in case of invalid VideoSourceToken.

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 **GetImagingSettings** request with an invalid VideoSourceToken.
4. The DUT returns **env:Sender/ter:InvalidArgVal/ter:NoSource** SOAP 1.2 fault.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- 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: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition.

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

Note: See Annex A.2 for Name and Token Parameters Length limitations.

5.1.5 IMAGING COMMAND GETOPTIONS – INVALID VIDEOSOURCETOKEN

Test Case ID: IMAGING-1-1-11

Specification Coverage: Get options

Feature under test: GetOptions

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of GetOptions command in case of invalid VideoSourceToken.

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 **GetOptions** request with an invalid VideoSourceToken.
4. The DUT returns **env:Sender/ter:InvalidArgVal/ter:NoSource** SOAP 1.2 fault.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- 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: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition.

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

Note: See Annex A.2 for Name and Token Parameters Length limitations.

5.1.6 IMAGING COMMAND SETIMAGINGSETTINGS – INVALID VIDEOSOURCETOKEN

Test Case ID: IMAGING-1-1-12

Specification Coverage: Set imaging settings

Feature under test: SetImagingSettings

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of SetImagingSettings command in case of invalid VideoSourceToken.

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 SetImagingSettingsRequest message with an invalid VideoSourceToken.
4. The DUT returns **env:Sender/ter:InvalidArgVal/ter:NoSource** SOAP 1.2 fault.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- 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: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition

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

Note: See Annex A.2 for Name and Token Parameters Length limitations.

5.1.7 IMAGING COMMAND SETIMAGINGSETTINGS

Test Case ID: IMAGING-1-1-14

Specification Coverage: Set imaging settings

Feature under test: SetImagingSettings

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behaviour of SetImagingSettings command and to verify DUT Imaging Setting.

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 **GetOptions** request (VideoSourceToken) to retrieve current imaging options for Video Source.
4. Verify the **GetOptionsResponse** message from the DUT.
5. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.
6. Verify the **GetImagingSettingsResponse** message from the DUT.
7. If the DUT did not return BacklightCompensation setting in the **GetOptionsResponse** message, then go to the step 26.
8. If the DUT did not return two BacklightCompensation.Mode settings (ON and OFF) in the **GetOptionsResponse** message, then go to the step 17.
9. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, BacklightCompensation.Mode = backlightCompensationMode1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.
10. Verify the SetImagingSettingsResponse message from the DUT.
11. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.
12. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that BacklightCompensation.Mode setting was applied.
13. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.
14. Verify the **SetImagingSettingsResponse** message from the DUT.
15. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.
16. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

17. If DUT did not return BacklightCompensation.Level settings with Max > Min or BacklightCompensation.Mode = ON setting in the **GetOptionsResponse** message, then go to the step 26.

18. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, BacklightCompensation.Mode = ON, BacklightCompensation.Level = backlightCompensationLevel1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

19. Verify the **SetImagingSettingsResponse** message from the DUT.

20. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

21. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that BacklightCompensation.Level setting was applied.

22. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

23. Verify the **SetImagingSettingsResponse** message from the DUT.

24. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

25. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

26. If the DUT did not return Brightness settings with Max > Min in the **GetOptionsResponse** message, then go to the step 35.

27. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Brightness = brightness1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

28. Verify the **SetImagingSettingsResponse** message from the DUT.

29. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

30. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Brightness setting was applied.

31. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

32. Verify the **SetImagingSettingsResponse** message from the DUT.

33. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

34. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

35. If the DUT did not return ColorSaturation settings with Max > Min in the **GetOptionsResponse** message, then go to the step 44.

36. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, ColorSaturation = colorSaturation1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

37. Verify the **SetImagingSettingsResponse** message from the DUT.

38. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

39. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that ColorSaturation setting was applied.

40. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

41. Verify the **SetImagingSettingsResponse** message from the DUT.

42. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

43. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

44. If the DUT did not return Contrast settings with Max > Min in the **GetOptionsResponse** message then go to the step 53.

45. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Contrast = contrast1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

46. Verify the **SetImagingSettingsResponse** message from the DUT.

47. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

48. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Contrast setting was applied.

49. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

50. Verify the **SetImagingSettingsResponse** message from the DUT.

51. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

52. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

53. If the DUT did not return Sharpness settings with Max > Min in the **GetOptionsResponse** message, then go to the step 62.

54. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Sharpness = sharpness1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

55. Verify the **SetImagingSettingsResponse** message from the DUT.

56. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

57. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Sharpness setting was applied.

58. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

59. Verify the **SetImagingSettingsResponse** message from the DUT.

60. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

61. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

62. If the DUT did not return Exposure setting in the **GetOptionsResponse** message, then go to the step 164.

63. If the DUT did not return two Exposure.Mode settings (AUTO and MANUAL) in the **GetOptionsResponse** message, then go to the step 72.

64. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Exposure.Mode = exposureMode1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

65. Verify the **SetImagingSettingsResponse** message from the DUT.

66. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

67. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Exposure.Mode setting was applied.

68. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

69. Verify the **SetImagingSettingsResponse** message from the DUT.

70. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

71. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

72. If the DUT did not return Exposure.Mode = AUTO setting in the **GetOptionsResponse** message, then go to the step 136.

73. If the DUT did not return Exposure.Priority settings with Max > Min in the **GetOptionsResponse** message, then go to the step ???.

74. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Exposure.Mode = AUTO, Exposure.Priority = exposurePriority1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

75. Verify the **SetImagingSettingsResponse** message from the DUT.

76. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

77. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Exposure.Priority settings were applied.

78. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

79. Verify the **SetImagingSettingsResponse** message from the DUT.

80. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

81. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

82. If the DUT did not return Exposure.MinExposureTime settings with Max > Min in the **GetOptionsResponse** message, then go to the step 91.

83. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Exposure.Mode = AUTO, Exposure.MinExposureTime = exposureMinExposureTime1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

84. Verify the **SetImagingSettingsResponse** message from the DUT.

85. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

86. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Exposure.MinExposureTime setting was applied.

87. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

88. Verify the **SetImagingSettingsResponse** message from the DUT.

89. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

90. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

91. If the DUT did not return Exposure.MaxExposureTime settings with Max > Min in the **GetOptionsResponse** message, then go to the step 100.

92. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Exposure.Mode = AUTO, Exposure.MaxExposureTime = exposureMaxExposureTime1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

93. Verify the **SetImagingSettingsResponse** message from the DUT.

94. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

95. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Exposure.MaxExposureTime setting was applied.

96. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

97. Verify the **SetImagingSettingsResponse** message from the DUT.

98. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

99. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

100. If the DUT did not return Exposure.MinGain settings with Max > Min in the **GetOptionsResponse** message, then go to the step 109.

101. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Exposure.Mode = AUTO, Exposure.MinGain = exposureMinGain1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

102. Verify the **SetImagingSettingsResponse** message from the DUT.

103. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

104. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Exposure.MinGain setting was applied.

105. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

106. Verify the **SetImagingSettingsResponse** message from the DUT.

107. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

108. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

109. If the DUT did not return Exposure.MaxGain settings with Max > Min in the **GetOptionsResponse** message, then go to the step 118.
110. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Exposure.Mode = AUTO, Exposure.MaxGain = exposureMaxGain1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.
111. Verify the **SetImagingSettingsResponse** message from the DUT.
112. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.
113. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Exposure.MaxGain setting was applied.
114. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.
115. Verify the **SetImagingSettingsResponse** message from the DUT.
116. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.
117. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.
118. If the DUT did not return Exposure.MinIris settings with Max > Min in the **GetOptionsResponse** message, then go to the step 127.
119. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Exposure.Mode = AUTO, Exposure.MinIris = exposureMinIris1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.
120. Verify the **SetImagingSettingsResponse** message from the DUT.
121. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.
122. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Exposure.MinIris setting was applied.
123. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

124. Verify the **SetImagingSettingsResponse** message from the DUT.

125. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

126. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

127. If the DUT did not return Exposure.MaxIris settings with Max > Min in the **GetOptionsResponse** message, then go to the step 136.

128. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Exposure.Mode = AUTO, Exposure.MaxIris = exposureMaxIris1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

129. Verify the **SetImagingSettingsResponse** message from the DUT.

130. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

131. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Exposure.MaxIris setting was applied.

132. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

133. Verify the **SetImagingSettingsResponse** message from the DUT.

134. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

135. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

136. If the DUT did not return Exposure.Mode = MANUAL setting in the **GetOptionsResponse** message, then go to the step 164.

137. If the DUT did not return Exposure.ExposureTime settings with Max > Min in the **GetOptionsResponse** message, then go to the step 146.

138. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Exposure.Mode = MANUAL, Exposure.ExposureTime = exposureExposureTime1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

139. Verify the **SetImagingSettingsResponse** message from the DUT.

140. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

141. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Exposure.ExposureTime settings were applied.

142. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

143. Verify the **SetImagingSettingsResponse** message from the DUT.

144. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

145. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

146. If the DUT did not return Exposure.Gain settings with Max > Min in the **GetOptionsResponse** message, then go to the step 155.

147. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Exposure.Mode = MANUAL, Exposure.Gain = exposureGain1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

148. Verify the **SetImagingSettingsResponse** message from the DUT.

149. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

150. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Exposure.Gain settings were applied.

151. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

152. Verify the **SetImagingSettingsResponse** message from the DUT.

153. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

154. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

155. If the DUT did not return Exposure.Iris settings with Max > Min in the **GetOptionsResponse** message, then go to the step 164.

156. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Exposure.Mode = MANUAL, Exposure.Iris = exposureIris1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

157. Verify the **SetImagingSettingsResponse** message from the DUT.

158. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

159. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Exposure.Iris settings were applied.

160. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

161. Verify the **SetImagingSettingsResponse** message from the DUT.

162. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

163. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

164. If the DUT did not return Focus setting in the **GetOptionsResponse** message, then go to the step 202.

165. If the DUT did not return at least two Focus.AutoFocusModes settings (AUTO or MANUAL) in the **GetOptionsResponse** message, then go to the step 174.

166. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Focus.AutoFocusMode = focusAutoFocusMode1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

167. Verify the **SetImagingSettingsResponse** message from the DUT.

168. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

169. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Focus.AutoFocusMode setting was applied.

170. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

171. Verify the **SetImagingSettingsResponse** message from the DUT.

172. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

173. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

174. If the DUT did not return Focus.AutoFocusModes = AUTO setting in the **GetOptionsResponse** message, then go to the step 193.

175. If the DUT did not return Focus.NearLimit settings with Max > Min in the **GetOptionsResponse** message, then go to the step 184.

176. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Focus.AutoFocusModes = AUTO, Focus.NearLimit = focusNearLimit1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

177. Verify the **SetImagingSettingsResponse** message from the DUT.

178. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

179. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Focus.NearLimit setting was applied.

180. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

181. Verify the **SetImagingSettingsResponse** message from the DUT.

182. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

183. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

184. If the DUT did not return Focus.FarLimit settings with Max > Min in the **GetOptionsResponse** message, then go to the step 193.

185. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Focus.AutoFocusModes = AUTO, Focus.FarLimit = focusFarLimit1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

186. Verify the **SetImagingSettingsResponse** message from the DUT.

187. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

188. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Focus.FarLimit setting was applied.

189. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

190. Verify the **SetImagingSettingsResponse** message from the DUT.

191. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

192. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

193. If the DUT did not return Focus.AutoFocusModes = MANUAL setting or Focus.DefaultSpeed settings with Max > Min in the **GetOptionsResponse** message, then go to the step 202.

194. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Focus.AutoFocusModes = MANUAL, Focus.DefaultSpeed = focusDefaultSpeed1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

195. Verify the **SetImagingSettingsResponse** message from the DUT.

196. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

197. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Focus.DefaultSpeed setting was applied.

198. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

199. Verify the **SetImagingSettingsResponse** message from the DUT.

200. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

201. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

202. If the DUT did not return at least two IrCutFilterModes settings in the **GetOptionsResponse** message, then go to the step 211.

203. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, IrCutFilter = irCutFilter1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

204. Verify the **SetImagingSettingsResponse** message from the DUT.

205. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

206. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that IrCutFilter setting was applied.

207. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

208. Verify the **SetImagingSettingsResponse** message from the DUT.

209. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging **settings**.

210. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

211. If the DUT did not return WhiteBalance setting in the **GetOptionsResponse** message, then go to the step 240.

212. If the DUT did not return at least two WhiteBalance.Mode settings (AUTO or MANUAL) in the **GetOptionsResponse** message, then go to the step 221.

213. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, WhiteBalance.Mode = whiteBalanceMode1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

214. Verify the **SetImagingSettingsResponse** message from the DUT.

215. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

216. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that WhiteBalance.Mode setting was applied.

217. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

218. Verify the **SetImagingSettingsResponse** message from the DUT.

219. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

220. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

221. If the DUT did not return WhiteBalance.Mode = MANUAL setting in the **GetOptionsResponse** message, then go to the step 240.

222. If DUT did not return WhiteBalance.YrGain settings with Max > Min in the **GetOptionsResponse** message, then go to the step 231.

223. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, WhiteBalance.Mode = MANUAL, WhiteBalance.CrGain = whiteBalanceCrGain1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

224. Verify the **SetImagingSettingsResponse** message from the DUT.

225. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

226. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that WhiteBalance.CrGain setting was applied.

227. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

228. Verify the **SetImagingSettingsResponse** message from the DUT.

229. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

230. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

231. If the DUT did not return WhiteBalance.YbGain settings with Max > Min in the **GetOptionsResponse** message, then go to the step 240.

232. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, WhiteBalance.Mode = MANUAL, WhiteBalance.CbGain = whiteBalanceCbGain1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

233. Verify the **SetImagingSettingsResponse** message from the DUT.

234. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

235. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that WhiteBalance.CbGain setting was applied.

236. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

237. Verify the **SetImagingSettingsResponse** message from the DUT.

238. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

239. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

240. If the DUT did not return WideDynamicRange setting in the **GetOptionsResponse** message, then go to the step 259.

241. If the DUT did not return two WideDynamicRange.Mode settings (ON and OFF) in the **GetOptionsResponse** message, then go to the step 250.

242. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, WideDynamicRange.Mode = wideDynamicRangeMode1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

243. Verify the **SetImagingSettingsResponse** message from the DUT.

244. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

245. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that WideDynamicRange.Mode setting was applied.

246. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

247. Verify the **SetImagingSettingsResponse** message from the DUT.

248. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

249. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

250. If the DUT did not return WideDynamicRange.Level settings with Max > Min or WideDynamicRange.Mode = ON setting in the **GetOptionsResponse** message, then go to the step 259.

251. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, WideDynamicRange.Mode = ON, WideDynamicRange.Level = wideDynamicRangeLevel1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

252. Verify the **SetImagingSettingsResponse** message from the DUT.

253. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

254. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that WideDynamicRange.Level setting was applied.

255. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

256. Verify the **SetImagingSettingsResponse** message from the DUT.

257. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

258. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

259. If the DUT did not return Extension.ImageStabilization settings in the **GetOptionsResponse** message, then go to the step 274.

260. If the DUT did not return at least two Extension.ImageStabilization.Mode settings in the **GetOptionsResponse** message, then go to the step 269.

261. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Extension.ImageStabilization.Mode = extensionImageStabilizationMode1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

262. Verify the **SetImagingSettingsResponse** message from the DUT.

263. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

264. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Extension.ImageStabilization.Mode setting was applied.

265. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

266. Verify the **SetImagingSettingsResponse** message from the DUT.

267. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

268. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that changed settings were returned to initial state.

269. If the DUT did not return Extension.ImageStabilization.Level settings with Max > Min or Extension.ImageStabilization.Mode = ON setting in the **GetOptionsResponse** message, then go to the step 274.

270. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Extension.ImageStabilization.Mode = ON, Extension.ImageStabilization.Level = extensionImageStabilizationLevel1 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

271. Verify the **SetImagingSettingsResponse** message from the DUT.

272. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

273. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that Extension.ImageStabilization.Level setting was applied.

274. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken) to restore old imaging settings.

275. Verify the **SetImagingSettingsResponse** message from the DUT.

276. ONVIF Client will repeat steps 3-275 for other video sources supported by the DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetOptionsResponse** message.
- The DUT did not send a valid **GetOptionsResponse** message.
- The DUT did not send **SetImagingSettings** message.
- The DUT did not send a valid **SetImagingSettings** message.
- The DUT did not send **GetImagingSettings** message.
- The DUT did not send a valid **GetImagingSettings** message.
- The DUT did not modify imaging settings by invoked **SetImagingSettings** request.
- The DUT did not return changed settings to initial state by invoked **SetImagingSettings** request.

Note: All imaging settings in each **SetImagingSettings** request shall have values from the range provided in **GetOptionsResponse** message. If options for imaging setting were skipped in **GetOptionsResponse** message, then they shall be skipped in **SetImagingSettings** request.

Note: At step 203 the ONVIF test tool selects irCutFilter1 = OFF if it is supported by the DUT and if it is not a current IrCutFilterMode.

5.1.8 IMAGING COMMAND SETIMAGINGSETTINGS ADDITIONAL FEATURES

Test Case ID: IMAGING-1-1-15

Specification Coverage: Set imaging settings

Feature under test: SetImagingSettings

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behaviour of DUT SetImagingSettings additional features.

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 **GetOptions** request (VideoSourceToken) to retrieve current imaging options for Video Source.
4. The DUT responds with **GetOptionsResponse** message with parameters
   - ImagingOptions =: *imagingOptions*
5. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.
6. Verify the **GetImagingSettingsResponse** message from the DUT.
7. If the DUT did not return at least two IrCutFilterAutoAdjustment.BoundaryType settings in the **GetOptionsResponse** message, then go to the next Parameter check.
8. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, IrCutFilterAutoAdjustment.BoundaryType = (Correct value from IrCutFilterAutoAdjustmentOptions, other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.
9. ONVIF Client will invoke **SetImagingSettings** request to set new imaging settings with the following parameters:
   - VideoSourceToken
   - IrCutFilterAutoAdjustment[0].BoundaryType := Correct value from IrCutFilterAutoAdjustmentOptions. If initial settings on the DUT do not contain all supported Boundary Types, then select value differs from values in initial settings.
   - IrCutFilterAutoAdjustment[0].BoundaryOffset := from +1.0 to -1.0. (other than current) if IrCutFilterAutoAdjustmentOptions.BoundaryOffset = true, otherwise skipped.

- IrCutFilterAutoAdjustment[0].ResponseTime := Correct value from IrCutFilterAutoAdjustmentOptions
- IrCutFilterAutoAdjustment[<other items>] skipped
- Other imaging parameters from **GetImagingSettingsResponse** message

10. Verify the **SetImagingSettingsResponse** message from the DUT.
11. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.
12. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that IrCutFilterAutoAdjustment.BoundaryType setting was applied.
13. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.
14. Verify the **SetImagingSettingsResponse** message from the DUT.
15. If the DUT did not return IrCutFilterAutoAdjustment.BoundaryOffset is true in the **GetOptionsResponse** message, then go to the next Parameter check.
16. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, IrCutFilterAutoAdjustment.BoundaryOffset = from +1.0 to -1.0. (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.
17. Verify the **SetImagingSettingsResponse** message from the DUT.
18. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.
19. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that IrCutFilterAutoAdjustment.BoundaryOffset setting was applied.
20. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.
21. Verify the **SetImagingSettingsResponse** message from the DUT.
22. If the DUT did not return IrCutFilterAutoAdjustment.ResponseTimeRange setting with Max>Min in the **GetOptionsResponse** message, then go to the next Parameter check.

23. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, IrCutFilterAutoAdjustment.ResponseTime = (Correct value from IrCutFilterAutoAdjustmentOptions, other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

24. Verify the **SetImagingSettingsResponse** message from the DUT.

25. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

26. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that IrCutFilterAutoAdjustment.ResponseTime setting was applied.

27. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

28. Verify the **SetImagingSettingsResponse** message from the DUT.

29. If the DUT did not return at least two ToneCompensation.Mode setting in the **GetOptionsResponse** message, then go to the next Parameter check.

30. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, ToneCompensation.Mode = enable/disable or automatic (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

31. Verify the **SetImagingSettingsResponse** message from the DUT.

32. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

33. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that setting was applied.

34. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

35. Verify the **SetImagingSettingsResponse** message from the DUT.

36. If the DUT did not return ToneCompensation.Level is true in the **GetOptionsResponse** message, then go to the next Parameter check.

37. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, ToneCompensation.Mode=on, ToneCompensation.Level = value from 0.0 to +1.0 (other

than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

38. Verify the **SetImagingSettingsResponse** message from the DUT.
39. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.
40. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that setting was applied.
41. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.
42. Verify the **SetImagingSettingsResponse** message from the DUT.
43. If the DUT did not return at least two Defogging.Mode settings in the **GetOptionsResponse** message, then go to the next Parameter check.
44. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Defogging.Mode = On/off or Auto (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.
45. Verify the **SetImagingSettingsResponse** message from the DUT.
46. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.
47. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that setting was applied.
48. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.
49. Verify the **SetImagingSettingsResponse** message from the DUT.
50. If the DUT did not return Defogging.Level is true in the **GetOptionsResponse** message, then go to the next Parameter check.
51. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, Defogging.Mode=On, Defogging.Level = value from 0.0 to +1.0 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.
52. Verify the **SetImagingSettingsResponse** message from the DUT.

53. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

54. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that setting was applied.

55. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.

56. Verify the **SetImagingSettingsResponse** message from the DUT.

57. If the DUT did not return NoiseReduction.Level is true in the **GetOptionsResponse** message, then go to the next Parameter check.

58. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, NoiseReduction.Level = value from 0.0 to +1.0 (other than current), other imaging parameters from **GetImagingSettingsResponse** message) to set new imaging settings.

59. Verify the **SetImagingSettingsResponse** message from the DUT.

60. ONVIF Client will invoke **GetImagingSettings** request (VideoSourceToken) to retrieve current imaging settings.

61. Verify the **GetImagingSettingsResponse** message from the DUT. Verify that setting was applied.

62. If *imagingOptions* contains Focus.Extension.@AFModes

    - ONVIF Client invokes **SetImagingSettings** request with parameters

        - VideoSourceToken := token of video source

        - Focus.AutoFocusMode = "AUTO"

        - Focus.AFMode = *imagingOptions*.Focus.Extension.@AFModes[0]

        - Other imaging parameters from **GetImagingSettingsResponse** message

    - The DUT responds with **SetImagingSettingsResponse**

    - ONVIF Client invokes **GetImagingSettings** request with parameters

        - VideoSourceToken := token of video source

    - The DUT responds with **GetImagingSettingsResponse** with parameters

        - ImagingSettings := *imagingSettings*

- If *imagingSettings*.Focus.AutoFocusMode != "AUTO", FAIL the test.
- If *imagingSettings*.Focus.AFMode != *imagingOptions*.Focus.Extension.@AFModes[0], FAIL the test.

63. ONVIF Client will invoke **SetImagingSettings** request (VideoSourceToken, all imaging parameters from **GetImagingSettingsResponse** message at step 6) to return imaging settings to initial state.
64. Verify the **SetImagingSettingsResponse** message from the DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- DUT did not send **GetVideoSourcesResponse** message.
- DUT did not send **GetProfilesResponse** message.

5.1.9 GET IMAGING SETTINGS AND GET OPTIONS CONSISTENCY

Test Case ID: IMAGING-1-1-16

Specification Coverage: Get imaging settings, Get options

Feature under test: GetImagingSettings, GetOptions

WSDL Reference: imaging.wsdl

Test Purpose: To verify retrieving of DUT Imaging Configuration using GetImagingSettings command. To verify retrieving of DUT Imaging Configuration Options using GetOptions command. To verify that all Configurations are consistent with Configuration Options.

Pre-Requisite: None

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. ONVIF Client retrieves Video Sources list by following the procedure mentioned in Section A.1, “Get Video Sources” with the following input and output parameters

   - out *videoSourcesList* - Video Sources list

4. For each Video Source token *videoSourceToken* from *videoSourcesList* repeat the following steps:

   4.1. ONVIF Client invokes **GetImagingSettings** request with parameters:

   - VideoSourceToken := *videoSourceToken*

   4.2. The DUT responds with **GetImagingSettingsResponse** message with parameters:

   - Configuration =: *configuration*

   4.3. ONVIF Client will invoke **GetOptions** request with parameters:

   - VideoSourceToken := *videoSourceToken*

   4.4. The DUT responds with **GetConfigurationOptionsResponse** message with parameters:

   - ConfigurationOptions =: *configurationOptions*

   4.5. If *configuration* contains BacklightCompensation element:

   4.5.1. If *configuration*.BacklightCompensation.Mode is not equal to one of *configurationOptions*.BacklightCompensation.Mode items, FAIL the test and skip other steps.

   4.5.2. If *configuration* contains BacklightCompensation.Level element:

   4.5.2.1. If *configuration*.BacklightCompensation.Level < *configurationOptions*.BacklightCompensation.Level.Min, FAIL the test and skip other steps.

   4.5.2.2. If *configuration*.BacklightCompensation.Level > *configurationOptions*.BacklightCompensation.Level.Max, FAIL the test and skip other steps.

   4.6. If *configuration* contains Brightness element:

   4.6.1. If *configuration*.Brightness < *configurationOptions*.Brightness.Min, FAIL the test and skip other steps.

4.6.2. If *configuration*.Brightness > *configurationOptions*.Brightness.Max, FAIL the test and skip other steps.

4.7. If *configuration* contains ColorSaturation element:

4.7.1. If *configuration*.ColorSaturation < *configurationOptions*.ColorSaturation.Min, FAIL the test and skip other steps.

4.7.2. If *configuration*.ColorSaturation > *configurationOptions*.ColorSaturation.Max, FAIL the test and skip other steps.

4.8. If *configuration* contains Contrast element:

4.8.1. If *configuration*.Contrast < *configurationOptions*.Contrast.Min, FAIL the test and skip other steps.

4.8.2. If *configuration*.Contrast > *configurationOptions*.Contrast.Max, FAIL the test and skip other steps.

4.9. If *configuration* contains Exposure element:

4.9.1. If *configuration*.Exposure.Mode is not equal to one of *configurationOptions*.Exposure.Mode items, FAIL the test and skip other steps.

4.9.2. If *configuration* contains Exposure.Priority element and *configuration*.Exposure.Priority is not equal to one of *configurationOptions*.Exposure.Priority items, FAIL the test and skip other steps.

4.9.3. If *configuration* contains Exposure.MinExposureTime element:

4.9.3.1. If *configuration*.Exposure.MinExposureTime < *configurationOptions*.Exposure.MinExposureTime.Min, FAIL the test and skip other steps.

4.9.3.2. If *configuration*.Exposure.MinExposureTime > *configurationOptions*.Exposure.MinExposureTime.Max, FAIL the test and skip other steps.

4.9.4. If *configuration* contains Exposure.MaxExposureTime element:

4.9.4.1. If *configuration*.Exposure.MaxExposureTime < *configurationOptions*.Exposure.MaxExposureTime.Min, FAIL the test and skip other steps.

4.9.4.2. If *configuration*.Exposure.MaxExposureTime > *configurationOptions*.Exposure.MaxExposureTime.Max, FAIL the test and skip other steps.

4.9.5. If *configuration* contains Exposure.MinGain element:

4.9.5.1. If *configuration*.Exposure.MinGain < *configurationOptions*.Exposure.MinGain.Min, FAIL the test and skip other steps.

4.9.5.2. If *configuration*.Exposure.MinGain > *configurationOptions*.Exposure.MinGain.Max, FAIL the test and skip other steps.

4.9.6. If *configuration* contains Exposure.MaxGain element:

4.9.6.1. If *configuration*.Exposure.MaxGain < *configurationOptions*.Exposure.MaxGain.Min, FAIL the test and skip other steps.

4.9.6.2. If *configuration*.Exposure.MaxGain > *configurationOptions*.Exposure.MaxGain.Max, FAIL the test and skip other steps.

4.9.7. If *configuration* contains Exposure.MinIris element:

4.9.7.1. If *configuration*.Exposure.MinIris < *configurationOptions*.Exposure.MinIris.Min, FAIL the test and skip other steps.

4.9.7.2. If *configuration*.Exposure.MinIris > *configurationOptions*.Exposure.MinIris.Max, FAIL the test and skip other steps.

4.9.8. If *configuration* contains Exposure.MaxIris element:

4.9.8.1. If *configuration*.Exposure.MaxIris < *configurationOptions*.Exposure.MaxIris.Min, FAIL the test and skip other steps.

4.9.8.2. If *configuration*.Exposure.MaxIris > *configurationOptions*.Exposure.MaxIris.Max, FAIL the test and skip other steps.

4.9.9. If *configuration* contains Exposure.ExposureTime element:

4.9.9.1. If *configuration*.Exposure.ExposureTime < *configurationOptions*.Exposure.ExposureTime.Min, FAIL the test and skip other steps.

4.9.9.2. If *configuration*.Exposure.ExposureTime > *configurationOptions*.Exposure.ExposureTime.Max, FAIL the test and skip other steps.

4.9.10. If *configuration* contains Exposure.Gain element:

4.9.10.1. If *configuration*.Exposure.Gain < *configurationOptions*.Exposure.Gain.Min, FAIL the test and skip other steps.

4.9.10.2. If *configuration*.Exposure.Gain > *configurationOptions*.Exposure.Gain.Max, FAIL the test and skip other steps.

4.9.11. If *configuration* contains Exposure.Iris element:

4.9.11.1. If *configuration*.Exposure.Iris < *configurationOptions*.Exposure.Iris.Min, FAIL the test and skip other steps.

4.9.11.2. If *configuration*.Exposure.Iris > *configurationOptions*.Exposure.Iris.Max, FAIL the test and skip other steps.

4.10. If *configuration* contains Focus element:

4.10.1. If *configuration*.Focus.AutoFocusMode is not equal to one of *configurationOptions*.Focus.AutoFocusModes items, FAIL the test and skip other steps.

4.10.2. If *configuration* contains Focus.DefaultSpeed element:

4.10.2.1. If *configuration*.Focus.DefaultSpeed < *configurationOptions*.Focus.DefaultSpeed.Min, FAIL the test and skip other steps.

4.10.2.2. If *configuration*.Focus.DefaultSpeed > *configurationOptions*.Focus.DefaultSpeed.Max, FAIL the test and skip other steps.

4.10.3. If *configuration* contains Focus.NearLimit element:

4.10.3.1. If *configuration*.Focus.NearLimit < *configurationOptions*.Focus.NearLimit.Min, FAIL the test and skip other steps.

4.10.3.2. If *configuration*.Focus.NearLimit > *configurationOptions*.Focus.NearLimit.Max, FAIL the test and skip other steps.

4.10.4. If *configuration* contains Focus.FarLimit element:

4.10.4.1. If *configuration*.Focus.FarLimit < *configurationOptions*.Focus.FarLimit.Min, FAIL the test and skip other steps.

4.10.4.2. If *configuration*.Focus.FarLimit > *configurationOptions*.Focus.FarLimit.Max, FAIL the test and skip other steps.

4.10.5. If *configuration*.Focus contains @AFMode list

4.10.5.1. If *configurationOptions* does not contain Focus.Extension.AFModes items, FAIL the test and skip other steps.

4.11. If *configuration* contains IrCutFilter element and *configuration*.IrCutFilter is not equal to one of *configurationOptions*.IrCutFilterModes items, FAIL the test and skip other steps.

4.12. If *configuration* contains Sharpness element:

4.12.1. If *configuration*.Sharpness < *configurationOptions*.Sharpness.Min, FAIL the test and skip other steps.

4.12.2. If *configuration*.Sharpness > *configurationOptions*.Sharpness.Max, FAIL the test and skip other steps.

4.13. If *configuration* contains WideDynamicRange element:

4.13.1. If *configuration*.WideDynamicRange.Mode is not equal to one of *configurationOptions*.WideDynamicRange.Mode items, FAIL the test and skip other steps.

4.13.2. If *configuration* contains WideDynamicRange.Level element:

4.13.2.1. If *configuration*.WideDynamicRange.Level < *configurationOptions*.WideDynamicRange.Level.Min, FAIL the test and skip other steps.

4.13.2.2. If *configuration*.WideDynamicRange.Level > *configurationOptions*.WideDynamicRange.Level.Max, FAIL the test and skip other steps.

4.14. If *configuration* contains WhiteBalance element:

4.14.1. If *configuration*.WhiteBalance.Mode is not equal to one of *configurationOptions*.WhiteBalance.Mode items, FAIL the test and skip other steps.

4.14.2. If *configuration* contains WhiteBalance.CrGain element:

4.14.2.1. If *configuration*.WhiteBalance.CrGain < *configurationOptions*.WhiteBalance.YrGain.Min, FAIL the test and skip other steps.

4.14.2.2. If *configuration*.WhiteBalance.CrGain > *configurationOptions*.WhiteBalance.YrGain.Max, FAIL the test and skip other steps.

4.14.3. If *configuration* contains WhiteBalance.CbGain element:

4.14.3.1. If *configuration*.WhiteBalance.CbGain < *configurationOptions*.WhiteBalance.YbGain.Min, FAIL the test and skip other steps.

4.14.3.2. If *configuration*.WhiteBalance.CbGain > *configurationOptions*.WhiteBalance.YbGain.Max, FAIL the test and skip other steps.

4.15. If *configuration* contains Extension.ImageStabilization element:

4.15.1. If *configuration*.Extension.ImageStabilization.Mode is not equal to one of *configurationOptions*.Extension.ImageStabilization.Mode items, FAIL the test and skip other steps.

4.15.2. If *configuration* contains Extension.ImageStabilization.Level element:

4.15.2.1. If *configuration*.Extension.ImageStabilization.Level < *configurationOptions*.Extension.ImageStabilization.Level.Min, FAIL the test and skip other steps.

4.15.2.2. If *configuration*.Extension.ImageStabilization.Level > *configurationOptions*.Extension.ImageStabilization.Level.Max, FAIL the test and skip other steps.

4.16. If *configuration* contains Extension.Extension.IrCutFilterAutoAdjustment element:

4.16.1. For each IrCutFilterAutoAdjustment (*irCutFilterAutoAdjustment*) in *configuration*.Extension.Extension repeat the following steps:

4.16.1.1. If *irCutFilterAutoAdjustment*.BoundaryType is not equal to one of *configurationOptions*.Extension.Extension.IrCutFilterAutoAdjustment.BoundaryTyp items, FAIL the test and skip other steps.

4.16.1.2. If *configuration* contains Extension.Extension.IrCutFilterAutoAdjustment.BoundaryOffset element and *configurationOptions*.Extension.Extension.IrCutFilterAutoAdjustment.BoundaryOffs is not equal to "True", FAIL the test and skip other steps.

4.16.1.3. If *configuration* contains Extension.Extension.IrCutFilterAutoAdjustment.ResponseTime element:

4.16.1.3.1. If *irCutFilterAutoAdjustment*.ResponseTime < *configurationOptions*.Extension.Extension.IrCutFilterAutoAdjustment.I FAIL the test and skip other steps.

4.16.1.3.2. If *irCutFilterAutoAdjustment*.ResponseTime > *configurationOptions*.Extension.Extension.IrCutFilterAutoAdjustment.I FAIL the test and skip other steps.

4.17. If *configuration* contains Extension.Extension.Extension.ToneCompensation element:

4.17.1. If *configuration*.Extension.Extension.Extension.ToneCompensation.Mode is not equal to one of *configurationOptions*.Extension.Extension.Extension.ToneCompensationOptions.Mode items, FAIL the test and skip other steps.

4.17.2. If *configuration* contains Extension.Extension.Extension.ToneCompensation.Level element and *configurationOptions*.Extension.Extension.Extension.ToneCompensationOptions.Level is not equal to "True", FAIL the test and skip other steps.

4.18. If *configuration* contains Extension.Extension.Extension.Defogging element:

4.18.1. If *configuration*.Extension.Extension.Extension.Defogging.Mode is not equal to one of *configurationOptions*.Extension.Extension.Extension.DefoggingOptions.Mode items, FAIL the test and skip other steps.

4.18.2. If *configuration* contains Extension.Extension.Extension.Defogging.Level element and *configurationOptions*.Extension.Extension.Extension.DefoggingOptions.Level is not equal to "True", FAIL the test and skip other steps.

4.19. If *configuration* contains Extension.Extension.Extension.NoiseReduction element and *configurationOptions*.Extension.Extension.Extension.NoiseReductionOptions.Level is not equal to "True", FAIL the test and skip other steps.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- DUT did not send **GetImagingSettingsResponse** message.
- DUT did not send **GetConfigurationOptionsResponse** message.

5.2 Focus Move

5.2.1 IMAGING COMMAND GETMOVEOPTIONS

Test Case ID: IMAGING-2-1-1

Specification Coverage: Get move options

Feature under test: GetMoveOptions

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of GetMoveOptions command.

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 **GetMoveOptions** request (VideoSourceToken) to retrieve current move options for Video Source.
4. Verify the **GetMoveOptionsResponse** message from the DUT.
5. ONVIF Client will repeat steps 3-4 for other video sources supported by the DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetMoveOptionsResponse** message.
- The DUT did not send valid **GetMoveOptionsResponse** message.
- The DUT returned **GetMoveOptionsResponse** message with Min value greater than Max for at least one setting defined by Min and Max boundary:
  - Absolute.Position
  - Absolute.Speed
  - Relative.Distance
  - Relative.Speed
  - Continuous.Speed

Note: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition

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

5.2.2 IMAGING COMMAND ABSOLUTE MOVE

Test Case ID: IMAGING-2-1-3

Specification Coverage: Move

Feature under test: Move

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of Move command in case of Absolute Move.

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 **GetMoveOptions** request (VideoSourceToken) to retrieve current move options for Video Source.
4. Verify the **GetMoveOptionsResponse** message from the DUT.
5. If Absolute Move is not supported by Video Source (there is no Absolute tag in **GetMoveOptionsResponse** message), go to step 10.
6. ONVIF Client will invoke **Move** request (VideoSourceToken, Absolute.Position = ["x1"], no Absolute.Speed) to move focus to position.
7. Verify that the DUT has returned a valid **MoveResponse** message.
8. If Absolute.Speed is supported by Video Source (there is tag Absolute.Speed in **GetMoveOptionsResponse** message), ONVIF Client will invoke **Move** request (VideoSourceToken, Absolute.Position = ["x2"], Absolute.Speed = ["v1"]) to move focus to position with the specified speed, else go to the step 10.
9. Verify that the DUT has returned a valid **MoveResponse** message.
10. ONVIF Client will repeat steps 3-9 for other video sources supported by the DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetMoveOptionsResponse** message.
- The DUT did not send a valid **GetMoveOptionsResponse** message.
- The DUT did not send **MoveResponse** message.
- The DUT did not send a valid **MoveResponse** message.

5.2.3 IMAGING COMMAND ABSOLUTE MOVE – INVALID SETTINGS

Test Case ID: IMAGING-2-1-4

Specification Coverage: Move

Feature under test: Move

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of Move command in case of Absolute Move and invalid settings.

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 **GetMoveOptions** request (VideoSourceToken) to retrieve current move options for Video Source.
4. Verify the **GetMoveOptionsResponse** message from the DUT.
5. If Absolute Move is not supported by Video Source (there is no Absolute tag in **GetMoveOptionsResponse** message), go to step 10.
6. ONVIF Client will invoke **Move** request (VideoSourceToken, Absolute.Position = invalid position, no Absolute.Speed).
7. The DUT will generate a SOAP 1.2 fault message.
8. If Absolute.Speed is supported by Video Source (there is tag Absolute.Speed in **GetMoveOptionsResponse** message) ONVIF Client will invoke **Move** request

(VideoSourceToken, Absolute.Position = ["x1"], Absolute.Speed = invalid speed), else go to the step 10.

9. The DUT will generate a SOAP 1.2 fault message.

10. ONVIF Client will repeat steps 5-9 for other video sources supported by DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetMoveOptionsResponse** message.
- The DUT did not send a valid **GetMoveOptionsResponse** message.
- The DUT did not send SOAP 1.2 fault message for **Move** request.
- The DUT sent an incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

Note: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition

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

5.2.4 IMAGING COMMAND RELATIVE MOVE

Test Case ID: IMAGING-2-1-5

Specification Coverage: Move

Feature under test: Move

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of Move command in case of Relative Move.

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 **GetMoveOptions** request (VideoSourceToken) to retrieve current move options for Video Source.
4. Verify the **GetMoveOptionsResponse** message from the DUT.
5. If Relative Move is not supported by Video Source (there is no Relative tag in **GetMoveOptionsResponse** message) go to step 10.
6. ONVIF Client will invoke **Move** request (VideoSourceToken, Relative.Distance = ["d1"], no Relative.Speed) to move focus to a distance.
7. Verify that the DUT has returned a valid **MoveResponse** message.
8. If Relative.Speed is supported by Video Source (there is tag Relative.Speed in **GetMoveOptionsResponse** message) ONVIF Client will invoke **Move** request (VideoSourceToken, Relative.Distance = ["d2"], Relative.Speed = ["v1"]) to move focus to a distance with specified speed, else go to the step 10.
9. Verify that the DUT has returned a valid **MoveResponse** message.
10. ONVIF Client will repeat steps 3-9 for other video sources supported by the DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetMoveOptionsResponse** message.
- The DUT did not send a valid **GetMoveOptionsResponse** message.
- The DUT did not send **MoveResponse** message.
- The DUT did not send a valid **MoveResponse** message.

5.2.5 IMAGING COMMAND RELATIVE MOVE – INVALID SETTINGS

Test Case ID: IMAGING-2-1-6

Specification Coverage: Move

Feature under test: Move

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of Move command in case of Relative Move and invalid settings.

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 **GetMoveOptions** request (VideoSourceToken) to retrieve current move options for Video Source.
4. Verify the **GetMoveOptionsResponse** message from the DUT.
5. If Relative Move is not supported by Video Source (there is no Relative tag in **GetMoveOptionsResponse** message) go to step 10.
6. ONVIF Client will invoke **Move** request (VideoSourceToken, Relative.Distance = invalid distance, no Relative.Speed).
7. The DUT will generate a SOAP 1.2 fault message.
8. If Relative.Speed is supported by Video Source (there is tag Relative.Speed in **GetMoveOptionsResponse** message) ONVIF Client will invoke **Move** request (VideoSourceToken, Relative.Distance = ["d1"], Relative.Speed = invalid speed), else go to the step 10.
9. The DUT will generate a SOAP 1.2 fault message.
10. ONVIF Client will repeat steps 3-9 for other video sources supported by DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetMoveOptionsResponse** message.
- The DUT did not send a valid **GetMoveOptionsResponse** message.
- The DUT did not send SOAP 1.2 fault message for **Move** request.
- The DUT sent an incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

Note: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition.

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

5.2.6 IMAGING COMMAND CONTINUOUS MOVE

Test Case ID: IMAGING-2-1-7

Specification Coverage: Move

Feature under test: Move, Stop

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of Move command in case of Continuous Move.

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 **GetMoveOptions** request (VideoSourceToken) to retrieve current move options for Video Source.
4. Verify the **GetMoveOptionsResponse** message from the DUT.
5. If Continuous Move is not supported by Video Source (there is no Continuous tag in **GetMoveOptionsResponse** message), go to step 10.
6. ONVIF Client will invoke **Move** request (VideoSourceToken, Continuous.Speed = ["v1"]) to start focus moving.
7. Verify that the DUT has returned a valid **MoveResponse** message.

8. ONVIF Client will invoke **Stop** request (VideoSourceToken) to stop focus moving.

9. Verify that the DUT has returned a valid **StopResponse** message.

10. ONVIF Client will repeat steps 3-10 for other video sources supported by the DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetMoveOptionsResponse** message.
- The DUT did not send a valid **GetMoveOptionsResponse** message.
- The DUT did not send **MoveResponse** message.
- The DUT did not send a valid **MoveResponse** message.
- The DUT did not send **StopResponse** message.
- The DUT did not send a valid **StopResponse** message.

Note: If Stop command is not supported by Video Source, ONVIF Client will use timeout to wait until the DUT stops moving.

5.2.7 IMAGING COMMAND CONTINUOUS MOVE – INVALID SETTINGS

Test Case ID: IMAGING-2-1-8

Specification Coverage: Move

Feature under test: Move

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of Move command in case of Continuous Move and invalid settings.

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 **GetMoveOptions** request (VideoSourceToken) to retrieve current move options for Video Source.
4. Verify the **GetMoveOptionsResponse** message from the DUT.
5. If Continuous Move is not supported by Video Source (there is no Continuous tag in **GetMoveOptionsResponse** message), go to step 8.
6. ONVIF Client will invoke **Move** request (VideoSourceToken, Continuous.Speed = invalid speed).
7. The DUT will generate a SOAP 1.2 fault message.
8. ONVIF Client will repeat steps 3-7 for other video sources supported by DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetMoveOptionsResponse** message.
- The DUT did not send a valid **GetMoveOptionsResponse** message.
- The DUT did not send SOAP 1.2 fault message for **Move** request.
- The DUT sent an incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

Note: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition.

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

5.2.8 IMAGING COMMAND MOVE – UNSUPPORTED MOVE

Test Case ID: IMAGING-2-1-10

Specification Coverage: Move

Feature under test: Move

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of Move command in case of unsupported move request.

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 **GetMoveOptions** request (VideoSourceToken) to retrieve current move options for Video Source.
4. Verify the **GetMoveOptionsResponse** message from the DUT.
5. If Absolute Move is supported by Video Source (there is Absolute tag in **GetMoveOptionsResponse** message), go to step 8.
6. ONVIF Client will invoke **Move** request (VideoSourceToken, Absolute.Position = any value, Absolute.Speed = any value).
7. The DUT will generate a SOAP 1.2 fault message.
8. If Relative Move is supported by Video Source (there is Relative tag in **GetMoveOptionsResponse** message), go to step 11.
9. ONVIF Client will invoke **Move** request (VideoSourceToken, Relative.Distance = any value, Relative.Speed = any value).
10. The DUT will generate a SOAP 1.2 fault message.
11. If Continuous Move is supported by Video Source (there is Continuous tag in **GetMoveOptionsResponse** message), go to step 14.
12. ONVIF Client will invoke **Move** request (VideoSourceToken, Continuous.Speed = any value).
13. The DUT will generate a SOAP 1.2 fault message.
14. ONVIF Client will repeat steps 3-13 for other video sources supported by DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetMoveOptionsResponse** message.
- The DUT did not send a valid **GetMoveOptionsResponse** message.
- The DUT did not send SOAP 1.2 fault message for **Move** request.
- The DUT sent an incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

Note: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition.

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

5.2.9 IMAGING COMMAND GETSTATUS

Test Case ID: IMAGING-2-1-11

Specification Coverage: Get imaging status

Feature under test: GetStatus

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of GetStatus command.

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 **GetStatus** request (VideoSourceToken) to retrieve current status for Video Source.
4. If the DUT generates a SOAP 1.2 fault message (**env:Receiver/ter:ActionNotSupported/ ter:NoImagingForSource** (in case Imaging is not supported for Video Source)

or **env:Receiver/ter:ActionNotSupported** (in case the GetStatus command is not implemented)), verify it.

5. If the DUT returns status for Video Source in the **GetStatusResponse** message, verify the **GetStatusResponse** message from the DUT.
6. ONVIF Client will repeat steps 3-5 for other video sources supported by the DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetStatusResponse** message or send SOAP 1.2 fault message.
- The DUT sent an incorrect SOAP 1.2 fault message (fault code, namespace, etc.).
- The DUT did not send **GetStatusResponse** message or send SOAP 1.2 fault message.
- The DUT sent an invalid **GetStatusResponse** message.

Note: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition.

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

5.2.10 IMAGING COMMAND STOP

Test Case ID: IMAGING-2-1-13

Specification Coverage: Stop

Feature under test: Stop

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of Stop command.

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 **Stop** request (VideoSourceToken) to stop movement for Video Source.
4. If the DUT generates a SOAP 1.2 fault message (**env:Receiver/ter:ActionNotSupported/ ter:NoImagingForSource** (in case Imaging is not supported for Video Source) or **env:Receiver/ter:ActionNotSupported** (in case the Stop command is not implemented)), verify it.
5. If the DUT returns the **StopResponse** message, verify the **StopResponse** message from the DUT.
6. ONVIF Client will repeat steps 3-5 for other video sources supported by the DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send **StopResponse** message or send SOAP 1.2 fault message.
- The DUT sent an incorrect SOAP 1.2 fault message (fault code, namespace, etc.).
- The DUT sent an invalid **StopResponse** message.

Note: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition.

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

5.2.11 IMAGING COMMAND GETMOVEOPTIONS – INVALID VIDEOSOURCETOKEN

Test Case ID: IMAGING-2-1-15

Specification Coverage: Get move options

Feature under test: GetMoveOptions

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of GetMoveOptions command in case of invalid VideoSourceToken.

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 **GetMoveOptions** request with an invalid VideoSourceToken.
4. The DUT returns **env:Sender/ter:InvalidArgVal/ter:NoSource** SOAP 1.2 fault.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- 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: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition

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

Note: See Annex A.2 for Name and Token Parameters Length limitations.

5.2.12 IMAGING COMMAND MOVE – INVALID VIDEOSOURCETOKEN

Test Case ID: IMAGING-2-1-16

Specification Coverage: Move

Feature under test: Move

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of Move command in case of invalid VideoSourceToken.

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 **Move** request with an invalid VideoSourceToken.
4. The DUT returns **env:Sender/ter:InvalidArgVal/ter:NoSource** SOAP 1.2 fault.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- 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: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition.

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

Note: See Annex A.2 for Name and Token Parameters Length limitations.

5.2.13 IMAGING COMMAND GETSTATUS – INVALID VIDEOSOURCETOKEN

Test Case ID: IMAGING-2-1-17

Specification Coverage: Get imaging status

Feature under test: GetStatus

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of GetStatus command in case of invalid VideoSourceToken.

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 **GetStatus** request with an invalid VideoSourceToken.
4. The DUT will generate a SOAP 1.2 fault message (**env:Sender/ter:InvalidArgVal/ter:NoSource** or **env:Receiver/ter:ActionNotSupported** (for the case when GetStatus command is not implemented)).

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- 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: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition.

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

Note: See Annex A.2 for Name and Token Parameters Length limitations.

5.2.14 IMAGING COMMAND STOP – INVALID VIDEOSOURCETOKEN

Test Case ID: IMAGING-2-1-18

Specification Coverage: Stop

Feature under test: Stop

WSDL Reference: imaging.wsdl

Test Purpose: To verify the behavior of Stop command in case of invalid VideoSourceToken.

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 **Stop** request with an invalid VideoSourceToken.
4. The DUT will generate a SOAP 1.2 fault message (**env:Sender/ter:InvalidArgVal/ter:NoSource** or **env:Receiver/ter:ActionNotSupported** (for the case when Stop command is not implemented)).

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- 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: See Annex in [ONVIF Base Test] for invalid SOAP 1.2 fault message definition.

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

Note: See Annex A.2 for Name and Token Parameters Length limitations.

5.3 Capabilities

5.3.1 IMAGING SERVICE CAPABILITIES

Test Case ID: IMAGING-3-1-1

Specification Coverage: Capability exchange

Feature under test: GetServiceCapabilities (for Imaging Service)

WSDL Reference: imaging.wsdl

Test Purpose: To verify Imaging Service Capabilities of the DUT.

Pre-Requisite: Imaging Service was received from the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke **GetServiceCapabilities** request to retrieve Imaging service capabilities of the DUT.
4. Verify the **GetServiceCapabilitiesResponse** message from the DUT.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- The DUT did not send valid **GetServiceCapabilitiesResponse** message.

5.3.2 GET SERVICES AND GET IMAGING SERVICE CAPABILITIES CONSISTANCY

Test Case ID: IMAGING-3-1-2

Specification Coverage: Capability exchange

Feature under test: GetServices, GetServiceCapabilities (for Imaging Service)

WSDL Reference: devicemgmt.wsdl, imaging.wsdl

Test Purpose: To verify Get Services and Imaging 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 **GetServices** request (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 **GetServiceCapabilities** request to retrieve Imaging service capabilities of the DUT.

6. Verify the **GetServiceCapabilitiesResponse** message from the DUT.

Test Result:

PASS –

- 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 Imaging service if it will have Namespace element that is equal to "http://www.onvif.org/ver20/imaging/wsdl".

5.4 Events

5.4.1 REALTIME PULLPOINT SUBSCRIPTION – IMAGE TOO BLURRY

Test Case ID: IMAGING-4-1-1

Specification Coverage: Events (ONVIF Imaging Service Specification), Tampering (ONVIF Imaging Service Specification), ImageTooBlurry (ONVIF Imaging Service Specification), Topic Filter (ONVIF Core Specification)

Feature under test: GetEventProperties, CreatePullPointSubscription, PullMessages

WSDL Reference: event.wsdl

Test Purpose: To verify tns1:VideoSource/ImageTooBlurry/ImagingService, tns1:VideoSource/ImageTooBlurry/AnalyticsService, and tns1:VideoSource/ImageTooBlurry/RecordingService event format. To verify event generation for tns1:VideoSource/ImageTooBlurry/ImagingService, tns1:VideoSource/ImageTooBlurry/AnalyticsService, and tns1:VideoSource/ImageTooBlurry/RecordingService.

Pre-Requisite: Device supports Image Too Blurry feature. 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 invokes **GetEventProperties** request.
4. The DUT responds with a **GetEventPropertiesResponse** message with parameters
   - TopicNamespaceLocation list
   - FixedTopicSet
   - TopicSet =: *topicSet*
   - TopicExpressionDialect list
   - MessageContentFilterDialect list
   - MessageContentSchemaLocation list
5. If *topicSet* does not contain at least one of the following topic, FAIL the test and skip other steps:
   - tns1:VideoSource/ImageTooBlurry/ImagingService
   - tns1:VideoSource/ImageTooBlurry/AnalyticsService
   - tns1:VideoSource/ImageTooBlurry/RecordingService
6. If *topicSet* contains **"tns1:VideoSource/ImageTooBlurry/ImagingService"** topic, ONVIF Client checks topic structure by following the procedure mentioned in Annex A.3 with the following input and output parameters
   - in *topicSet* – TopicSet
   - in **"tns1:VideoSource/ImageTooBlurry/ImagingService"** – topic to check

7. If *topicSet* contains **"tns1:VideoSource/ImageTooBlurry/AnalyticsService"** topic, ONVIF Client checks topic structure by following the procedure mentioned in Annex A.3 with the following input and output parameters

   - in *topicSet* – TopicSet
   - in **"tns1:VideoSource/ImageTooBlurry/AnalyticsService"** – topic to check

8. If *topicSet* contains **"tns1:VideoSource/ImageTooBlurry/RecordingService"** topic, ONVIF Client checks topic structure by following the procedure mentioned in Annex A.3 with the following input and output parameters

   - in *topicSet* – TopicSet
   - in **"tns1:VideoSource/ImageTooBlurry/RecordingService"** – topic to check

9. ONVIF Client invokes **CreatePullPointSubscription** with parameters

   - Filter.TopicExpression := all topics listed at step 5 and present in *topicSet* separated with '|' symbol (example: "tns1:VideoSource/ImageTooBlurry/ImagingService| tns1:VideoSource/ImageTooBlurry/AnalyticsService|tns1:VideoSource/ImageTooBlurry/ RecordingService")
   - Filter.TopicExpression.@Dialect:= "http://www.onvif.org/ver10/tev/topicExpression/ ConcreteSet"

10. The DUT responds with a **CreatePullPointSubscriptionResponse** message with parameters

    - SubscriptionReference =: *s*
    - CurrentTime =: *ct*
    - TerminationTime =: *tt*

11. Until *timeout1* timeout expires, repeat the following steps:

    11.1. ONVIF Client waits for time *t* := min{(*tt*-*ct*)/2, 1 second}.

    11.2. ONVIF Client invokes **PullMessages** to the subscription endpoint s with parameters

    - Timeout := PT60S
    - MessageLimit := 1

    11.3. The DUT responds with **PullMessagesResponse** message with parameters

    - CurrentTime =: *ct*

- TerminationTime =: *tt*
- NotificationMessage list =: *notificationMessageList*

11.4. If *notificationMessageList* is not empty, ONVIF Client executes the following:

11.4.1. Set *notification* := the first Notification from *notificationMessageList*

11.4.2. If *notification*.Topic value is not equal to one of topics from Filter.TopicExpression from CreatePullPointSubscription request, FAIL the test and skip other steps.

11.4.3. If *notification*.Message.Message have PropertyOperation attribute with value is equal to "Initialized", ONVIF Client checks *notification* notification by following the procedure mentioned in Annex A.4 with the following input and output parameters.

- in *notification* – notification

11.5. If *topicSet* contains **"tns1:VideoSource/ImageTooBlurry/ImagingService"** topic and *timeout1* expires for step 12 without NotificationMessage with topic value equals to "tns1:VideoSource/ImageTooBlurry/ImagingService" and with PropertyOperation = "Initialized", FAIL the test and skip other steps.

11.6. If *topicSet* contains **"tns1:VideoSource/ImageTooBlurry/AnalyticsService"** topic and *timeout1* expires for step 12 without NotificationMessage with topic value equals to "tns1:VideoSource/ImageTooBlurry/AnalyticsService" and with PropertyOperation = "Initialized", FAIL the test and skip other steps.

11.7. If *topicSet* contains **"tns1:VideoSource/ImageTooBlurry/RecordingService"** topic and *timeout1* expires for step 12 without NotificationMessage with topic value equals to "tns1:VideoSource/ImageTooBlurry/RecordingService" and with PropertyOperation = "Initialized", FAIL the test and skip other steps.

12. ONVIF Client sends an **Unsubscribe** to the subscription endpoint s.

13. The DUT responds with **UnsubscribeResponse** message.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

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

Note: *timeout1* will be taken from Operation Delay field of ONVIF Device Test Tool.

5.4.2 REALTIME PULLPOINT SUBSCRIPTION – IMAGE TOO DARK

Test Case ID: IMAGING-4-1-2

Specification Coverage: Events (ONVIF Imaging Service Specification), Tampering (ONVIF Imaging Service Specification), ImageTooDark (ONVIF Imaging Service Specification), Topic Filter (ONVIF Core Specification)

Feature under test: GetEventProperties, CreatePullPointSubscription, PullMessages

WSDL Reference: event.wsdl

Test Purpose: To verify tns1:VideoSource/ImageTooDark/ImagingService, tns1:VideoSource/ImageTooDark/AnalyticsService, and tns1:VideoSource/ImageTooDark/RecordingService event format. To verify event generation for tns1:VideoSource/ImageTooDark/ImagingService, tns1:VideoSource/ImageTooDark/AnalyticsService, and tns1:VideoSource/ImageTooDark/RecordingService.

Pre-Requisite: Device supports Image Too Dark feature. 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 invokes **GetEventProperties** request.
4. The DUT responds with a **GetEventPropertiesResponse** message with parameters
    - TopicNamespaceLocation list
    - FixedTopicSet

- TopicSet =: *topicSet*
- TopicExpressionDialect list
- MessageContentFilterDialect list
- MessageContentSchemaLocation list

5. If *topicSet* does not contain at least one of the following topic, FAIL the test and skip other steps:
   - tns1:VideoSource/ImageTooDark/ImagingService
   - tns1:VideoSource/ImageTooDark/AnalyticsService
   - tns1:VideoSource/ImageTooDark/RecordingService
6. If *topicSet* contains **"tns1:VideoSource/ImageTooDark/ImagingService"** topic, ONVIF Client checks topic structure by following the procedure mentioned in Annex A.3 with the following input and output parameters
   - in *topicSet* – TopicSet
   - in **"tns1:VideoSource/ImageTooDark/ImagingService"** – topic to check
7. If *topicSet* contains **"tns1:VideoSource/ImageTooDark/AnalyticsService"** topic, ONVIF Client checks topic structure by following the procedure mentioned in Annex A.3 with the following input and output parameters
   - in *topicSet* – TopicSet
   - in **"tns1:VideoSource/ImageTooDark/AnalyticsService"** – topic to check
8. If *topicSet* contains **"tns1:VideoSource/ImageTooDark/RecordingService"** topic, ONVIF Client checks topic structure by following the procedure mentioned in Annex A.3 with the following input and output parameters
   - in *topicSet* – TopicSet
   - in **"tns1:VideoSource/ImageTooDark/RecordingService"** – topic to check
9. ONVIF Client invokes **CreatePullPointSubscription** with parameters
   - Filter.TopicExpression := all topics listed at step 5 and present in *topicSet* separated with '|' symbol (example: "tns1:VideoSource/ImageTooBlurry|tns1:VideoSource/ImageTooBlurry/ImagingService|tns1:VideoSource/ImageTooBlurry/AnalyticsService|tns1:VideoSource/ImageTooBlurry/RecordingService")

- Filter.TopicExpression.@Dialect:= "http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet"

10. The DUT responds with a **CreatePullPointSubscriptionResponse** message with parameters
    - SubscriptionReference =: *s*
    - CurrentTime =: *ct*
    - TerminationTime =: *tt*

11. Until *timeout1* timeout expires, repeat the following steps:

    11.1. ONVIF Client waits for time *t* := min{(*tt*-*ct*)/2, 1 second}.

    11.2. ONVIF Client invokes **PullMessages** to the subscription endpoint s with parameters
    - Timeout := PT60S
    - MessageLimit := 1

    11.3. The DUT responds with **PullMessagesResponse** message with parameters
    - CurrentTime =: *ct*
    - TerminationTime =: *tt*
    - NotificationMessage list =: *notificationMessageList*

    11.4. If *notificationMessageList* is not empty, ONVIF Client executes the following:

    11.4.1. Set *notification* := the first Notification from *notificationMessageList*

    11.4.2. If *notification*.Topic value is not equal to one of topics from Filter.TopicExpression from CreatePullPointSubscription request, FAIL the test and skip other steps.

    11.4.3. If *notification*.Message.Message have PropertyOperation attribute with value is equal to "Initialized",ONVIF Client checks *notification* notification by following the procedure mentioned in Annex A.4 with the following input and output parameters.
    - in *notification* – notification

    11.5. If *topicSet* contains **"tns1:VideoSource/ImageTooDark/ImagingService"** topic and *timeout1* expires for step 12 without NotificationMessage with topic value equals to "tns1:VideoSource/ImageTooDark/ImagingService" and with PropertyOperation = "Initialized", FAIL the test and skip other steps.

11.6. If *topicSet* contains **"tns1:VideoSource/ImageTooDark/AnalyticsService"** topic and *timeout1* expires for step 12 without NotificationMessage with topic value equals to "tns1:VideoSource/ImageTooDark/AnalyticsService" and with PropertyOperation = "Initialized", FAIL the test and skip other steps

11.7. If *topicSet* contains **"tns1:VideoSource/ImageTooDark/RecordingService"** topic and *timeout1* expires for step 12 without NotificationMessage with topic value equals to "tns1:VideoSource/ImageTooDark/RecordingService" and with PropertyOperation = "Initialized", FAIL the test and skip other steps.

12. ONVIF Client sends an **Unsubscribe** to the subscription endpoint *s*.

13. The DUT responds with **UnsubscribeResponse** message.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

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

Note: *timeout1* will be taken from Operation Delay field of ONVIF Device Test Tool.

5.4.3 REALTIME PULLPOINT SUBSCRIPTION – IMAGE TOO BRIGHT

Test Case ID: IMAGING-4-1-3

Specification Coverage: Events (ONVIF Imaging Service Specification), Tampering (ONVIF Imaging Service Specification), ImageTooBright (ONVIF Imaging Service Specification), Topic Filter (ONVIF Core Specification)

Feature under test: GetEventProperties, CreatePullPointSubscription, PullMessages

WSDL Reference: event.wsdl

Test Purpose: To verify tns1:VideoSource/ ImageTooBright/ImagingService, tns1:VideoSource/ ImageTooBright/AnalyticsService, and tns1:VideoSource/ImageTooBright/RecordingService event format. To verify event generation for tns1:VideoSource/ImageTooBright/ImagingService, tns1:VideoSource/ImageTooBright/AnalyticsService, and tns1:VideoSource/ImageTooBright/ RecordingService.

Pre-Requisite: Device supports Image Too Bright feature. 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 invokes **GetEventProperties** request.
4. The DUT responds with a **GetEventPropertiesResponse** message with parameters
   - TopicNamespaceLocation list
   - FixedTopicSet
   - TopicSet =: *topicSet*
   - TopicExpressionDialect list
   - MessageContentFilterDialect list
   - MessageContentSchemaLocation list
5. If *topicSet* does not contain at least one of the following topic, FAIL the test and skip other steps:
   - tns1:VideoSource/ImageTooBright/ImagingService
   - tns1:VideoSource/ImageTooBright/AnalyticsService
   - tns1:VideoSource/ImageTooBright/RecordingService
6. If *topicSet* contains **"tns1:VideoSource/ImageTooBright/ImagingService"** topic, ONVIF Client checks topic structure by following the procedure mentioned in Annex A.3 with the following input and output parameters
   - in *topicSet* – TopicSet
   - in **"tns1:VideoSource/ImageTooBright/ImagingService"** – topic to check

7. If *topicSet* contains **"tns1:VideoSource/ImageTooBright/AnalyticsService"** topic, ONVIF Client checks topic structure by following the procedure mentioned in Annex A.3 with the following input and output parameters
   - in *topicSet* – TopicSet
   - in **"tns1:VideoSource/ImageTooBright/AnalyticsService"** – topic to check
8. If *topicSet* contains **"tns1:VideoSource/ImageTooBright/RecordingService"** topic, ONVIF Client checks topic structure by following the procedure mentioned in Annex A.3 with the following input and output parameters
   - in *topicSet* – TopicSet
   - in **"tns1:VideoSource/ImageTooBright/RecordingService"** – topic to check
9. ONVIF Client invokes **CreatePullPointSubscription** with parameters
   - Filter.TopicExpression := all topics listed at step 5 and present in *topicSet* separated with '|' symbol (example: "tns1:VideoSource/ImageTooBlurry|tns1:VideoSource/ImageTooBlurry/ImagingService|tns1:VideoSource/ImageTooBlurry/AnalyticsService|tns1:VideoSource/ImageTooBlurry/RecordingService")
   - Filter.TopicExpression.@Dialect:= "http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet"
10. The DUT responds with a **CreatePullPointSubscriptionResponse** message with parameters
    - SubscriptionReference =: *s*
    - CurrentTime =: *ct*
    - TerminationTime =: *tt*
11. Until *timeout1* timeout expires, repeat the following steps:

    11.1. ONVIF Client waits for time *t* := min{(*tt*-*ct*)/2, 1 second}.

    11.2. ONVIF Client invokes **PullMessages** to the subscription endpoint s with parameters
    - Timeout := PT60S
    - MessageLimit := 1

    11.3. The DUT responds with **PullMessagesResponse** message with parameters
    - CurrentTime =: *ct*

- TerminationTime =: *tt*
- NotificationMessage list =: *notificationMessageList*

11.4. If *notificationMessageList* is not empty, ONVIF Client executes the following:

11.4.1. Set *notification* := the first Notification from *notificationMessageList*

11.4.2. If *notification*.Topic value is not equal to one of topics from Filter.TopicExpression from CreatePullPointSubscription request, FAIL the test and skip other steps.

11.4.3. If *notification*.Message.Message have PropertyOperation attribute with value is equal to "Initialized",ONVIF Client checks *notification* notification by following the procedure mentioned in Annex A.4 with the following input and output parameters.

- in *notification* – notification

11.5. If *topicSet* contains **"tns1:VideoSource/ImageTooBright/ImagingService"** topic and *timeout1* expires for step 12 without NotificationMessage with topic value equals to "tns1:VideoSource/ImageTooBright/ImagingService" and with PropertyOperation = "Initialized", FAIL the test and skip other steps.

11.6. If *topicSet* contains **"tns1:VideoSource/ImageTooBright/AnalyticsService"** topic and *timeout1* expires for step 12 without NotificationMessage with topic value equals to "tns1:VideoSource/ImageTooBright/AnalyticsService" and with PropertyOperation = "Initialized", FAIL the test and skip other steps.

11.7. If *topicSet* contains **"tns1:VideoSource/ImageTooBright/RecordingService"** topic and *timeout1* expires for step 12 without NotificationMessage with topic value equals to "tns1:VideoSource/ImageTooBright/RecordingService" and with PropertyOperation = "Initialized", FAIL the test and skip other steps.

12. ONVIF Client sends an **Unsubscribe** to the subscription endpoint *s*.

13. The DUT responds with **UnsubscribeResponse** message.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

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

Note: *timeout1* will be taken from Operation Delay field of ONVIF Device Test Tool.

5.4.4 REALTIME PULLPOINT SUBSCRIPTION – GLOBAL SCENE CHANGE

Test Case ID: IMAGING-4-1-4

Specification Coverage: Events (ONVIF Imaging Service Specification), Tampering (ONVIF Imaging Service Specification), GlobalSceneChange (ONVIF Imaging Service Specification), Topic Filter (ONVIF Core Specification)

Feature under test: GetEventProperties, CreatePullPointSubscription, PullMessages

WSDL Reference: event.wsdl

Test Purpose: To verify tns1:VideoSource/GlobalSceneChange/ImagingService, tns1:VideoSource/GlobalSceneChange/AnalyticsService, and tns1:VideoSource/GlobalSceneChange/RecordingService event format. To verify event generation for tns1:VideoSource/GlobalSceneChange/ImagingService, tns1:VideoSource/GlobalSceneChange/AnalyticsService, and tns1:VideoSource/GlobalSceneChange/RecordingService.

Pre-Requisite: Device supports Global Scene Change feature. 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 invokes **GetEventProperties** request.
4. The DUT responds with a **GetEventPropertiesResponse** message with parameters
    - TopicNamespaceLocation list
    - FixedTopicSet

- TopicSet =: *topicSet*
- TopicExpressionDialect list
- MessageContentFilterDialect list
- MessageContentSchemaLocation list

5. If *topicSet* does not contain at least one of the following topic, FAIL the test and skip other steps:

   - tns1:VideoSource/GlobalSceneChange/ImagingService
   - tns1:VideoSource/GlobalSceneChange/AnalyticsService
   - tns1:VideoSource/GlobalSceneChange/RecordingService

6. If *topicSet* contains **"tns1:VideoSource/GlobalSceneChange/ImagingService"** topic, ONVIF Client checks topic structure by following the procedure mentioned in Annex A.3 with the following input and output parameters

   - in *topicSet* – TopicSet
   - in **"tns1:VideoSource/GlobalSceneChange/ImagingService"** – topic to check

7. If *topicSet* contains **"tns1:VideoSource/GlobalSceneChange/AnalyticsService"** topic, ONVIF Client checks topic structure by following the procedure mentioned in Annex A.3 with the following input and output parameters

   - in *topicSet* – TopicSet
   - in **"tns1:VideoSource/GlobalSceneChange/AnalyticsService"** – topic to check

8. If *topicSet* contains **"tns1:VideoSource/GlobalSceneChange/RecordingService"** topic, ONVIF Client checks topic structure by following the procedure mentioned in Annex A.3 with the following input and output parameters

   - in *topicSet* – TopicSet
   - in **"tns1:VideoSource/GlobalSceneChange/RecordingService"** – topic to check

9. ONVIF Client invokes **CreatePullPointSubscription** with parameters

   - Filter.TopicExpression := all topics listed at step 5 and present in *topicSet* separated with '|' symbol (example: "tns1:VideoSource/ImageTooBlurry|tns1:VideoSource/ImageTooBlurry/ImagingService|tns1:VideoSource/ImageTooBlurry/AnalyticsService|tns1:VideoSource/ImageTooBlurry/RecordingService")

- Filter.TopicExpression.@Dialect:= "http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet"

10. The DUT responds with a **CreatePullPointSubscriptionResponse** message with parameters
    - SubscriptionReference =: *s*
    - CurrentTime =: *ct*
    - TerminationTime =: *tt*
11. Until *timeout1* timeout expires, repeat the following steps:

    11.1. ONVIF Client waits for time *t* := min{(*tt*-*ct*)/2, 1 second}.

    11.2. ONVIF Client invokes **PullMessages** to the subscription endpoint s with parameters
    - Timeout := PT60S
    - MessageLimit := 1

    11.3. The DUT responds with **PullMessagesResponse** message with parameters
    - CurrentTime =: *ct*
    - TerminationTime =: *tt*
    - NotificationMessage list =: *notificationMessageList*

    11.4. If *notificationMessageList* is not empty, ONVIF Client executes the following:

    11.4.1. Set *notification* := the first Notification from *notificationMessageList*

    11.4.2. If *notification*.Topic value is not equal to one of topics from Filter.TopicExpression from CreatePullPointSubscription request, FAIL the test and skip other steps.

    11.4.3. If *notification*.Message.Message have PropertyOperation attribute with value is equal to "Initialized",ONVIF Client checks *notification* notification by following the procedure mentioned in Annex A.4 with the following input and output parameters.
    - in *notification* – notification

    11.5. If *topicSet* contains **"tns1:VideoSource/GlobalSceneChange/ImagingService"** topic and *timeout1* expires for step 12 without NotificationMessage with topic value equals to "tns1:VideoSource/GlobalSceneChange/ImagingService" and with PropertyOperation = "Initialized", FAIL the test and skip other steps.

11.6. If *topicSet* contains **"tns1:VideoSource/GlobalSceneChange/AnalyticsService"** topic and *timeout1* expires for step 12 without NotificationMessage with topic value equals to "tns1:VideoSource/GlobalSceneChange/AnalyticsService" and with PropertyOperation = "Initialized", FAIL the test and skip other steps.

11.7. If *topicSet* contains **"tns1:VideoSource/GlobalSceneChange/RecordingService"** topic and *timeout1* expires for step 12 without NotificationMessage with topic value equals to "tns1:VideoSource/GlobalSceneChange/RecordingService" and with PropertyOperation = "Initialized", FAIL the test and skip other steps.

12. ONVIF Client sends an **Unsubscribe** to the subscription endpoint *s*.

13. The DUT responds with **UnsubscribeResponse** message.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

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

Note: *timeout1* will be taken from Operation Delay field of ONVIF Device Test Tool.

5.4.5 REALTIME PULLPOINT SUBSCRIPTION – MOTION ALARM

Test Case ID: IMAGING-4-1-5

Specification Coverage: Events (ONVIF Imaging Service Specification), MotionAlarm (ONVIF Imaging Service Specification), Topic Filter (ONVIF Core Specification)

Feature under test: GetEventProperties, CreatePullPointSubscription, PullMessages

WSDL Reference: event.wsdl

Test Purpose: To verify tns1:VideoSource/MotionAlarm, event format. To verify event generation for tns1:VideoSource/MotionAlarm.

Pre-Requisite: Device supports Motion Alarm feature. 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 invokes **GetEventProperties** request.
4. The DUT responds with a **GetEventPropertiesResponse** message with parameters
   - TopicNamespaceLocation list
   - FixedTopicSet
   - TopicSet =: *topicSet*
   - TopicExpressionDialect list
   - MessageContentFilterDialect list
   - MessageContentSchemaLocation list
5. If *topicSet* does not contain tns1:VideoSource/MotionAlarm topic, FAIL the test and skip other steps.
6. ONVIF Client checks topic structure by following the procedure mentioned in Annex A.3 with the following input and output parameters
   - in *topicSet* – TopicSet
   - in **"tns1:VideoSource/MotionAlarm"** – topic to check
7. ONVIF Client invokes **CreatePullPointSubscription** with parameters
   - Filter.TopicExpression := "tns1:VideoSource/MotionAlarm"
   - Filter.TopicExpression.@Dialect:= "http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet"
8. The DUT responds with a **CreatePullPointSubscriptionResponse** message with parameters
   - SubscriptionReference =: *s*

- CurrentTime =: *ct*
- TerminationTime =: *tt*

9. Until *timeout1* timeout expires, repeat the following steps:

   9.1. ONVIF Client waits for time *t* := min{(*tt*-*ct*)/2, 1 second}.

   9.2. ONVIF Client invokes **PullMessages** to the subscription endpoint s with parameters

   - Timeout := PT60S
   - MessageLimit := 1

   9.3. The DUT responds with **PullMessagesResponse** message with parameters

   - CurrentTime =: *ct*
   - TerminationTime =: *tt*
   - NotificationMessage list =: *notificationMessageList*

   9.4. If *notificationMessageList* is not empty, ONVIF Client executes the following:

   9.4.1. Set *notification* := the first Notification from *notificationMessageList*

   9.4.2. If *notification*.Topic value is not equal to tns1:VideoSource/MotionAlarm, FAIL the test and skip other steps.

   9.4.3. If *notification*.Message.Message have PropertyOperation attribute with value is equal to "Initialized",ONVIF Client checks *notification* notification by following the procedure mentioned in Annex A.4 with the following input and output parameters.

   - in *notification* – notification

   9.5. If *timeout1* expires for step 9 without NotificationMessage with topic value equals to "tns1:VideoSource/MotionAlarm" and with PropertyOperation = "Initialized", FAIL the test and skip other steps.

10. ONVIF Client sends an **Unsubscribe** to the subscription endpoint *s*.

11. The DUT responds with **UnsubscribeResponse** message.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

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

Note: *timeout1* will be taken from Operation Delay field of ONVIF Device Test Tool.

Annex A Helper Procedures and Additional Notes

A.1 Get Video Sources

Name: HelperGetAndValidateVideoSources

Procedure Purpose: Helper procedure to get list of video sources with Imaging support.

Pre-requisite: Device IO Service or Media Service is received from the DUT. Imaging Service is received from the DUT.

Input: None.

Returns: List of Video Sources with Imaging support (*vsList*).

Procedure:

1. If Device IO Service is supported by the DUT:

   1.1. ONVIF Client invokes **GetVideoSources** request for Device IO Service.

   1.2. The DUT responds with **GetVideoSourcesResponse** message with parameters

   - Token list =: *videoSourceTokenList*

   1.3. Go to step 5.

2. ONVIF Client invokes **GetVideoSources** request for Media Service.

3. The DUT responds with **GetVideoSourcesResponse** message with parameters

   - VideoSources list =: *videoSourcesList*

4. Set *videoSourceTokenList* := list of @token from *videoSourcesList*.

5. If *videoSourceTokenList* is empty, FAIL the test and skip other steps.

6. If *videoSourceTokenList* contains only one item, add *videoSourceTokenList*[0] to the *vsList* list and skip other procedure steps.

7. For each Video Source token *token* in *videoSourceTokenList* repeat the following steps:

   7.1. ONVIF Client invokes **GetOptions** request with parameters

   - VideoSourceToken := *token*

7.2. The DUT returns **env:Receiver/ter:ActionNotSupported/ter:NoImagingForSource** SOAP 1.2 fault or the DUT responds with **GetOptionsResponse** message with parameters

- ImagingOptions

7.3. If the DUT returns **GetOptionsResponse** message, add *token* to the *vsList* list.

8. If *vsList* is empty, FAIL the test and skip other steps.

Procedure Result:

PASS –

- DUT passes all assertions.

FAIL –

- DUT did not send **GetVideoSourcesResponse** message.
- DUT did not send **GetOptionsResponse** message or **env:Receiver/ter:ActionNotSupported/ter:NoImagingForSource** SOAP 1.2 fault.

A.2 Name and Token Parameters

There are following limitations on maximum length of Name and Token parameters that sell be used during tests by ONVIF Device Test Tool to prevent faults from DUT:

1. Name shall be less than or equal to 64 characters (only readable characters accepted).
2. Token shall be less than or equal to 64 characters (only readable characters accepted).
3. UTF-8 character set shall be used for Name and Token.

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

A.3 Check Imaging Topic structure in GetEventPropertiesResponse

Name: HelperCheckImagingTopic

Procedure Purpose: Helper procedure to check structure of Imaging Topic in GetEventPropertiesResponse.

Pre-requisite: None.

Input: TopicSet (*topicSet*), topic to check (*topic*).

Returns: None.

Procedure:

1. ONVIF Client verifies *topic* topic from *topicSet*:

   1.1. If *topic*.MessageDescription.IsProperty is not equal to true, FAIL the test and skip other steps.

   1.2. If *topic* does not contain MessageDescription.Source.SimpleItemDescription item with Name = "Source", FAIL the test and skip other steps.

   1.3. If *topic*.MessageDescription.Source.SimpleItemDescription with Name = "Source" does not have Type = "tt:ReferenceToken", FAIL the test and skip other steps.

   1.4. If *topic* does not contain MessageDescription.Data.SimpleItemDescription item with Name = "State", FAIL the test and skip other steps.

   1.5. If *topic*.MessageDescription.Data.SimpleItemDescription with Name = "State" does not have Type = "xs:boolean", FAIL the test and skip other steps.

A.4 Check Imaging Notification Message

Name: HelperCheckImagingNotification

Procedure Purpose: Helper procedure to check Imaging notification message.

Pre-requisite: None.

Input: Notification Message (*notification*).

Returns: None.

Procedure:

1. If *notification*.Message.Message does not have Source.SimpleItem with Name = "Source" and Value with type is equal to "tt:ReferenceToken", FAIL the test and skip other steps.

2. If *notification*.Message.Message does not have Data.SimpleItem with Name = "State" and Value with type is equal to "xs:boolean", FAIL the test and skip other steps.