

ONVIF[®]

Analytics Engine Device Test Specification

Version 19.12

December 2019

© 2019 ONVIF, Inc. All rights reserved.

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

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

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

REVISION HISTORY

Vers.	Date	Description
17.06	Mar 15, 2017	First Issue.
17.06	Apr 18, 2017	ANALYTICS-1-1-1 GET SUPPORTED RULES (MOTION REGION DETECTOR) added ANALYTICS-1-1-2 GET MOTION REGION DETECTOR RULE OPTIONS added
17.06	May 23, 2017	ANALYTICS-1-1-2 GET MOTION REGION DETECTOR RULE OPTIONS updated ANALYTICS-1-1-3 CREATE MOTION REGION DETECTOR RULE added
17.06	May 24, 2017	ANALYTICS-1-1-4 MODIFY MOTION REGION DETECTOR RULE added ANALYTICS-2-1-1 MOTION REGION DETECTOR EVENT added
17.12	Jul 24, 2017	The following test cases were changed according to #1444: ANALYTICS-1-1-2 GET MOTION REGION DETECTOR RULE OPTIONS ANALYTICS-1-1-4 MODIFY MOTION REGION DETECTOR RULE ANALYTICS-2-1-1 MOTION REGION DETECTOR EVENT
17.12	Jul 24, 2017	The following test cases were changed according to #1445: ANALYTICS-1-1-3 CREATE MOTION REGION DETECTOR RULE ANALYTICS-1-1-4 MODIFY MOTION REGION DETECTOR RULE ANALYTICS-2-1-1 MOTION REGION DETECTOR EVENT
17.12	Sept 18, 2017	The following test cases were added according to #1477: ANALYTICS-3-1-1 Get Services and Get Analytics Service Capabilities Consistency
17.12	Oct 16, 2017	Pre-Requisite of the following test cases were updated according to #1185: GET SUPPORTED RULES (MOTION REGION DETECTOR) GET MOTION REGION DETECTOR RULE OPTIONS CREATE MOTION REGION DETECTOR RULE MODIFY MOTION REGION DETECTOR RULE MOTION REGION DETECTOR EVENT
17.12	Nov 22, 2017	The following test case was updated according to #1184: ANALYTICS-2-1-1 MOTION REGION DETECTOR EVENT
17.12	Nov 28, 2017	The following test cases were updated according to #1398:

		<p>ANALYTICS-1-1-3 CREATE MOTION REGION DETECTOR RULE</p> <p>ANALYTICS-1-1-4 MODIFY MOTION REGION DETECTOR RULE</p> <p>ANALYTICS-2-1-1 MOTION REGION DETECTOR EVENT</p> <p>The following annexes were added according to #1398:</p> <p>Annex A.8 Calculate Free Space for Rule</p> <p>Annex A.9 Delete Rule with Requested Type</p> <p>The following test cases were updated according to #1185:</p> <p>ANALYTICS-1-1-3 CREATE MOTION REGION DETECTOR RULE</p> <p>ANALYTICS-1-1-4 MODIFY MOTION REGION DETECTOR RULE</p> <p>ANALYTICS-2-1-1 MOTION REGION DETECTOR EVENT</p>
18.06	Jan 22, 2018	<p>The following test cases were updated according to #1557:</p> <p>ANALYTICS-1-1-3 CREATE MOTION REGION DETECTOR RULE</p> <p>ANALYTICS-1-1-4 MODIFY MOTION REGION DETECTOR RULE</p> <p>ANALYTICS-2-1-1 MOTION REGION DETECTOR EVENT</p> <p>The following annexes was added:</p> <p>Annex A.10 Create Motion Region Detector Rule</p>
18.06	Jun 21, 2018	Reformatting document using new template
18.12	Aug 16, 2018	<p>The following test cases were changed according to #1709:</p> <p>ANALYTICS-1-1-3 CREATE MOTION REGION DETECTOR RULE</p> <p>ANALYTICS-1-1-4 MODIFY MOTION REGION DETECTOR RULE</p> <p>ANALYTICS-2-1-1 MOTION REGION DETECTOR EVENT</p> <p>Annex Create Motion Region Detector Rule</p>
18.12	Sept 05, 2018	<p>The following test cases were changed according to #1733:</p> <p>ANALYTICS-1-1-3 CREATE MOTION REGION DETECTOR RULE (Note about comparison of Armed field changed)</p> <p>ANALYTICS-1-1-4 MODIFY MOTION REGION DETECTOR RULE (Note about comparison of Armed field changed)</p>
18.12	Oct 08, 2018	<p>The following test cases were changed according to #1665:</p> <p>ANALYTICS-1-1-1 GET SUPPORTED RULES (MOTION REGION DETECTOR) (step 4.3.1 added)</p>
18.12	Dec 21, 2018	Switching Hub description in 'Network Configuration for DUT' section was updated according to #1737
18.12	Dec 27, 2018	<p>Check of @Type in GetRuleOptionsResponse was changed in the following test cases according to #1774:</p> <p>GET MOTION REGION DETECTOR RULE OPTIONS</p> <p>CREATE MOTION REGION DETECTOR RULE</p>

		<p>MODIFY MOTION REGION DETECTOR RULE</p> <p>MOTION REGION DETECTOR EVENT</p>
19.12	Apr 26, 2019	<p>The following was changed according to #1833:</p> <p>Scope (Analytics Engine section was replaced with the following sections: Motion Region Detector, Events, Capabilities)</p> <p>Scope (Analytics Modules section was added)</p> <p>Test Policy (Analytics Modules section was added)</p> <p>Analytics Engine (Analytics Modules section was added)</p>
19.12	Oct 18, 2019	<p>The following was changed according to #1831:</p> <p>Introduction (Analytics Modules section was updated)</p> <p>Test Policy (Analytics Modules section was updated)</p> <p>ANALYTICS-3-1-1 Get Services and Get Analytics Service Capabilities Consistency (updated with adding of Note)</p> <p>ANALYTICS-3-1-2 Analytics Service Capabilities (added)</p> <p>Annex A.13 Get Installed Analytics Modules (added)</p> <p>Annex A.14 Get Supported Metadata (added)</p>
19.12	Oct 23, 2019	<p>The following was changed according to #1837:</p> <p>ANALYTICS-4-1-5 Object Classification Metadata (added)</p>

Table of Contents

1	Introduction	8
1.1	Scope	8
1.2	Motion Region Detector	9
1.3	Events	9
1.4	Capabilities	9
1.5	Analytics Modules	9
2	Normative references	11
3	Terms and Definitions	13
3.1	Conventions	13
3.2	Definitions	13
3.3	Abbreviations	13
4	Test Overview	14
4.1	Test Setup	14
4.1.1	Network Configuration for DUT	14
4.2	Prerequisites	15
4.3	Test Policy	15
4.3.1	Motion Region Detector	15
4.3.2	Events	16
4.3.3	Capabilities	17
4.3.4	Analytics Modules	18
5	Analytics Engine	21
5.1	Motion Region Detector	21
5.1.1	GET SUPPORTED RULES (MOTION REGION DETECTOR)	21
5.1.2	GET MOTION REGION DETECTOR RULE OPTIONS	23
5.1.3	CREATE MOTION REGION DETECTOR RULE	24
5.1.4	MODIFY MOTION REGION DETECTOR RULE	30
5.2	Events	34
5.2.1	MOTION REGION DETECTOR EVENT	34
5.3	Capabilities	38
5.3.1	Get Services and Get Analytics Service Capabilities Consistency	38

5.3.2	Analytics Service Capabilities	40
5.4	Analytics Modules	41
5.4.1	Get Supported Analytics Modules	41
5.4.2	Get Analytics Module Options	43
5.4.3	Get Analytics Modules	45
5.4.4	Object Classification Metadata	47
A	Helper Procedures and Additional Notes	49
A.1	Get Analytics Configurations List	49
A.2	Get List of Analytics Configurations With Supporting of Required Rule Type	49
A.3	Get Specific Rule Options	50
A.4	Configure Media Profile with required Analytics Configuration	51
A.5	Get Rules	53
A.6	Create Pull Point Subscription	54
A.7	Delete Subscription	54
A.8	Calculate Free Space for Rule	55
A.9	Delete Rule with Requested Type	56
A.10	Create Motion Region Detector Rule	57
A.11	Topic Format Verification	60
A.12	Valid Topic Format	61
A.13	Get Installed Analytics Modules	61
A.14	Get Supported Metadata	62

1 Introduction

The goal of the ONVIF test specification set is to make it possible to realize fully interoperable IP physical security implementation from different vendors. The set of ONVIF test specification describes the test cases need to verify the [ONVIF Network Interface Specs] and [ONVIF Conformance] requirements. In addition, 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 Analytics Engine Test Specification acts as a supplementary document to the [ONVIF Network Interface Specs], illustrating test cases need to be executed and passed. And 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 Analytics Engine 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 to provide test cases to test individual requirements of ONVIF devices according to ONVIF Analytics service(s) 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.
- Poor streaming performance test (audio/video distortions, missing audio/video frames, incorrect lib synchronization etc.).

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 will cover its subset.

This ONVIF Analytics Engine Test Specification covers Analytics Service, which is a functional block of [ONVIF Network Interface Specs]. The following section gives a brief overview of each functional block and its scope.

1.2 Motion Region Detector

Motion Region Detector test cases cover verification of Rule interface for Motion Region Detector feature as mentioned in [ONVIF Analytics Spec]. It means that the following commands are covered by these test cases:

- GetSupportedRules (for Motion Region Detector Rule scope only);
- CreateRules (for Motion Region Detector Rule scope only);
- ModifyRules (for Motion Region Detector Rule scope only);
- DeleteRules (for Motion Region Detector Rule scope only);
- GetRules (for Motion Region Detector Rule scope only).

1.3 Events

Events test cases cover verification of property events defined in [ONVIF Analytics Spec]. Currently the following events are covered by these test cases:

- tns1:RuleEngine/MotionRegionDetector/Motion.

1.4 Capabilities

Capabilities test cases cover verification to get Analytics Service capabilities. It means that the following commands are covered by these test cases:

- GetServices (Analytics Service);
- GetServiceCapabilities.

1.5 Analytics Modules

Analytics Modules test cases cover verification of analytics modules configuration feature as mentioned in [ONVIF Analytics Spec]. It means that the following commands are covered by these test cases:

- GetSupportedAnalyticsModules;
- GetAnalyticsModules;
- GetAnalyticsModuleOptions.
- GetSupportedMetadata.

2 Normative references

- [ONVIF Conformance] ONVIF Conformance Process Specification:
<https://www.onvif.org/profiles/conformance/>
- [ONVIF Profile Policy] ONVIF Profile Policy:
<https://www.onvif.org/profiles/>
- [ONVIF Network Interface Specs] ONVIF Network Interface Specification documents:
<https://www.onvif.org/profiles/specifications/>
- [ONVIF Core Specs] ONVIF Core Specification:
<https://www.onvif.org/profiles/specifications/>
- [ONVIF Media2 Spec] ONVIF Media 2 Specification:
<https://www.onvif.org/profiles/specifications/>
- [ONVIF Analytics Spec] ONVIF Analytics Specification:
<https://www.onvif.org/profiles/specifications/>
- [ONVIF Base Test] ONVIF Base Device Test Specification:
<https://www.onvif.org/profiles/conformance/device-test/>
- [ISO/IEC Directives, Part 2] ISO/IEC Directives, Part 2, Annex H:
<http://www.iso.org/directives>
- [ISO 16484-5] ISO 16484-5:2014-09 Annex P:
<https://www.iso.org/obp/ui/#!iso:std:63753:en>
- [SOAP 1.2, Part 1] W3C SOAP 1.2, Part 1, Messaging Framework:
<http://www.w3.org/TR/soap12-part1/>
- [XML-Schema, Part 1] W3C XML Schema Part 1: Structures Second Edition:
<http://www.w3.org/TR/xmlschema-1/>
- [XML-Schema, Part 2] W3C XML Schema Part 2: Datatypes Second Edition:
<http://www.w3.org/TR/xmlschema-2/>

- [WS-Security] "Web Services Security: SOAP Message Security 1.1 (WS-Security 2004)", OASIS Standard, February 2006.:

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

3 Terms and Definitions

3.1 Conventions

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

3.2 Definitions

This section describes terms and definitions used in this document.

Profile	See ONVIF Profile Policy.
ONVIF Device	Computer appliance or software program that exposes one or multiple ONVIF Web Services.
ONVIF Client	Computer appliance or software program that uses ONVIF Web Services.
Media Profile	A media profile maps a video and/or audio source to a video and/or an audio encoder, PTZ and analytics configurations.
SOAP	SOAP is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. It uses XML technologies to define an extensible messaging framework providing a message construct that can be exchanged over a variety of underlying protocols.
Device Test Tool	ONVIF Device Test Tool that tests ONVIF Device implementation towards the ONVIF Test Specification set.
Video Analytics	Algorithms used to evaluate video data for meaning of content.
Audio Analytics	Algorithms used to evaluate audio data for meaning of content.

3.3 Abbreviations

This section describes abbreviations used in this document.

HTTP	Hyper Text Transport Protocol.
WSDL	Web Services Description Language.
XML	eXtensible Markup Language.
PTZ	Pan/Tilt/Zoom.

4 Test Overview

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

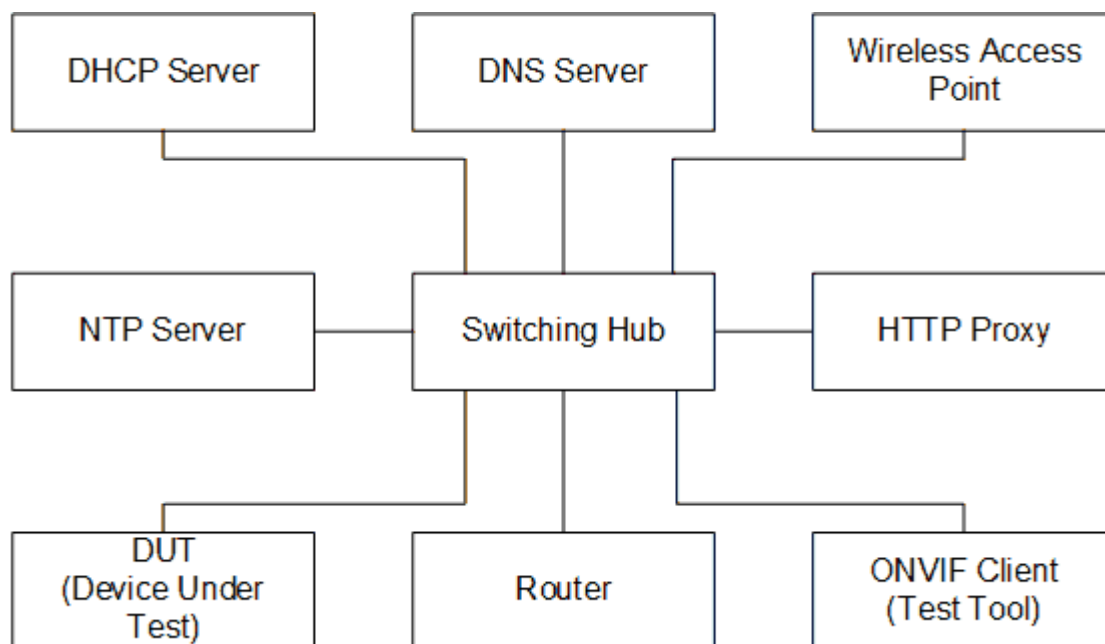
4.1 Test Setup

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

The test policies specific to the test case execution of Motion Region Detector functional block:

- DUT shall give the Analytics Service entry point by GetServices command, if DUT supports this service. Otherwise, these test cases will be skipped.
- DUT shall give the Media2 Service entry point by GetServices command, if DUT supports this service. Otherwise, these test cases will be skipped.
- DUT shall provide Motion Region Detector rule, if DUT supports this rule. Otherwise, these test cases will be skipped.

- DUT shall support the following commands:
 - GetServiceCapabilities
- If DUT returns RuleSupport capability as supported, then DUT shall support commands listed below. Otherwise, these test cases will be skipped.
 - GetServiceCapabilities
 - GetSupportedRules
 - GetRules
 - CreateRules
 - ModifyRules
 - DeleteRules
- If DUT returns RuleOptionsSupported capability as supported, then DUT shall support GetRuleOptions command. Otherwise, the following test cases will be skipped:
 - GET MOTION REGION DETECTOR RULE OPTIONS
 - MODIFY MOTION REGION DETECTOR RULE

Please, refer to [Section 5.1](#) for Motion Region Detector Test Cases.

4.3.2 Events

The test policies specific to the test case execution of Events functional block::

- DUT shall give the Analytics Service entry point and Event Service entry points by GetServices command, if DUT supports this service. Otherwise, these test cases will be skipped.
- DUT shall give the Media2 Service entry point by GetServices command, if DUT supports this service. Otherwise, these test cases will be skipped.
- DUT shall provide Motion Region Detector rule, if DUT supports this rule. Otherwise, these test cases will be skipped.
- DUT shall provide tns1:RuleEngine/MotionRegionDetector/Motion notification topic and Initialized event, if DUT supports Motion Region Detector rule. Otherwise, these test cases will be skipped.
- DUT shall support the following commands:

- GetServiceCapabilities
- GetEventProperties
- CreatePullPointSubscription
- PullMessages
- Unsubscribe
- If DUT returns RuleSupport capability as supported, then DUT shall support commands listed below. Otherwise, these test cases will be skipped.
 - GetServiceCapabilities
 - GetSupportedRules
 - CreateRules
 - DeleteRules
- If DUT returns RuleOptionsSupported capability as supported, then DUT shall support GetRuleOptions command. Otherwise, these test cases will be skipped.

Please, refer to [Section 5.2](#) for Motion Region Detector Test Cases.

4.3.3 Capabilities

The test policies specific to the test case execution of Capabilities functional block:

- DUT shall give the Analytics Service entry point by GetServices command, if DUT supports this service. Otherwise, these test cases will be skipped.
- DUT shall support the following commands:
 - GetServices
 - GetServiceCapabilities
- The following tests are performed
 - Getting capabilities with GetServiceCapabilities command
 - Getting capabilities with GetServices command

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

4.3.4 Analytics Modules

The test policies specific to the test case execution of Analytics Modules functional block:

- DUT shall give the Analytics Service entry point by GetServices command, if DUT supports this service. Otherwise, these test cases will be skipped.
- DUT shall support the following commands:
 - GetSupportedAnalyticsModules
 - GetAnalyticsModules
 - GetAnalyticsModuleOptions
- DUT shall support the following commands:
 - GetSupportedAnalyticsModules
 - GetAnalyticsModules
 - GetAnalyticsModuleOptions
- Additionally, DUT shall support the following commands which will be used as supplementary during the testing:
 - GetServices
- DUT shall return all supported analytics modules in GetSupportedAnalyticsModulesResponse response for the video analytics configuration specified in GetSupportedAnalyticsModules request.
- If DUT supports Supported Metadata as indicated by the Capabilities.SupportedMetadata, then DUT shall support the following commands
 - GetSupportedMetadata
- DUT shall indicate maximum number of analytics modules through the maxInstances attribute in GetSupportedAnalyticsModulesResponse.
- DUT shall return unique names of the parameters for each supported analytics modules in GetSupportedAnalyticsModulesResponse response.
- DUT shall return unique names of the messages for each supported analytics modules in GetSupportedAnalyticsModulesResponse response.
- DUT shall return valid parent topic value for each supported analytics modules in GetSupportedAnalyticsModulesResponse response.

- DUT shall return all configured analytics modules in `GetAnalyticsModulesResponse` response for the video analytics configuration specified in `GetAnalyticsModules` request.
- DUT shall return all analytics modules, which marked as fixed in `GetSupportedAnalyticsModulesResponse` response, in `GetAnalyticsModulesResponse` response for the video analytics configuration specified in `GetAnalyticsModules` request.
- If DUT shall return configured analytics modules in `GetAnalyticsModulesResponse` response with the structure defined in `GetSupportedAnalyticsModulesResponse` response for the corresponding analytics module type.
- If DUT supports receiving of analytics module options as indicated by `AnalyticsModuleOptionsSupported` capability:
 - DUT shall return options for all parameters of all supported analytics modules in `GetAnalyticsModuleOptionsResponse` response for the video analytics configuration specified in `GetAnalyticsModuleOptions` request, if `Type` is skipped.
 - DUT shall return options for all parameters in `GetAnalyticsModuleOptionsResponse` response for the video analytics configuration and supported analytics module specified in `GetAnalyticsModuleOptions` request.
 - DUT shall not return `RuleType` for any option in `GetAnalyticsModuleOptionsResponse` response.
- The following tests are performed:
 - Receiving of supported analytics modules for each video analytics configuration.
 - Receiving of all configured analytics modules for each video analytics configuration.
 - Verifying of consistency between configured analytics modules and supported analytics modules description.
 - If DUT supports receiving of analytics module options as indicated by `AnalyticsModuleOptionsSupported` capability:
 - Receiving of all analytics module options for each video analytics configuration.
 - Receiving of analytics module options for specified analytics module type for each video analytics configuration.
 - Verifying of consistency between analytics module options and supported analytics modules description.

- If DUT supports receiving of supported metadata as indicated by SupportedMetadata capability:
 - Receiving of supported metadata for each installed module type.

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

5 Analytics Engine

5.1 Motion Region Detector

5.1.1 GET SUPPORTED RULES (MOTION REGION DETECTOR)

Test Case ID: ANALYTICS-1-1-1

Specification Coverage: Get Supported rules (ONVIF Analytics Service Spec), Motion Region Detector (ONVIF Analytics Service Spec)

Feature Under Test: GetSupportedRules, RuleDescription for tt:MotionRegionDetector

WSDL Reference: analytics.wSDL, media2.wSDL

Test Purpose: To verify that device includes tt:MotionRegionDetector in GetSupportedRulesResponse. To verify structure of Motion Region Detector.

Pre-Requisite: Analytics Service is received from the DUT. Media2 Service is received from the DUT. Rule Engine is supported by the Device as indicated by the RuleSupport capabilities. Motion Region Detector Rule is supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client retrieves a list of Analytics Configurations by following the procedure mentioned in [Annex A.1](#) with the following input and output parameters
 - out *analyticsConfList* - a list of Analytics configurations
4. For each Analytics Configuration *analyticsConf* in *analyticsConfList* repeat the following steps:
 - 4.1. ONVIF Client invokes **GetSupportedRules** request with parameters
 - ConfigurationToken := *analyticsConf.@token*
 - 4.2. DUT responds with **GetSupportedRulesResponse** message with parameters
 - SupportedRules =: *supportedRules*

- 4.3. If *supportedRules* contains *RuleDescription* element (*motionRegionDetectorRuleDescription*) with *Name* value is equal to **tt:MotionRegionDetector**:
- 4.3.1. If *motionRegionDetectorRuleDescription* does not have *maxInstances* attribute, FAIL the test and skip other steps.
 - 4.3.2. If *motionRegionDetectorRuleDescription* does not have *Parameters.ElementItemDescription* element with *Name* attribute value is equal to "MotionRegion", FAIL the test and skip other steps.
 - 4.3.3. If *Type* attribute value is not equal to "axt:MotionRegionConfig" for *motionRegionDetectorRuleDescription.Parameters.ElementItemDescription* with *Name* attribute value is equal to "MotionRegion", FAIL the test and skip other steps.
 - 4.3.4. If *motionRegionDetectorRuleDescription* does not have *Messages.Source.SimpleItemDescription* element with *Name* attribute value is equal to "VideoSource", FAIL the test and skip other steps.
 - 4.3.5. If *Type* attribute value is not equal to "tt:ReferenceToken" for *motionRegionDetectorRuleDescription.Messages.Source.SimpleItemDescription* with *Name* attribute value is equal to "VideoSource", FAIL the test and skip other steps.
 - 4.3.6. If *motionRegionDetectorRuleDescription* does not have *Messages.Source.SimpleItemDescription* element with *Name* attribute value is equal to "RuleName", FAIL the test and skip other steps.
 - 4.3.7. If *Type* attribute value is not equal to "xs:string" for *motionRegionDetectorRuleDescription.Messages.Source.SimpleItemDescription* with *Name* attribute value is equal to "RuleName", FAIL the test and skip other steps.
 - 4.3.8. If *motionRegionDetectorRuleDescription* does not have *Messages.Data.SimpleItemDescription* element with *Name* attribute value is equal to "State", FAIL the test and skip other steps.
 - 4.3.9. If *Type* attribute value is not equal to "xs:boolean" for *motionRegionDetectorRuleDescription.Messages.Data.SimpleItemDescription* with *Name* attribute value is equal to "State", FAIL the test and skip other steps.

4.3.10. If `Messages.ParentTopic` value is not equal to `"tns1:RuleEngine/MotionRegionDetector/Motion"` for Messages with `Source.SimpleItemDescription.Name` value is equal to `VideoSource` and with `Source.SimpleItemDescription.Name` value is equal to `RuleName`, FAIL the test and skip other steps.

5. If there was no `RuleDescription` element with `Name` value is equal to **tt:MotionRegionDetector** in at least one *supportedRules* at step 4.2 [21], FAIL the test.

Test Result:**PASS –**

- DUT passes all assertions.

FAIL –

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

5.1.2 GET MOTION REGION DETECTOR RULE OPTIONS

Test Case ID: ANALYTICS-1-1-2

Specification Coverage: Get Rule Options (ONVIF Analytics Service Spec), Motion Region Detector (ONVIF Analytics Service Spec)

Feature Under Test: `GetRuleOptions`, `MotionRegionConfigOptions`

WSDL Reference: `analytics.wsdl`, `media2.wsdl`

Test Purpose: To verify retrieving of `MotionRegionConfigOptions` by `GetRuleOptions` operation.

Pre-Requisite: Analytics Service is received from the DUT. Media2 Service is received from the DUT. Rule Engine is supported by the Device as indicated by the `RuleSupport` capabilities, Rule Options is supported by the Device as indicated by the `RuleOptionsSupported` capabilities. Motion Region Detector Rule is supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

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

3. ONVIF Client retrieves a list of Analytics Configurations that supports `tt:MotionRegionDetector` type by following the procedure mentioned in [Annex A.2](#) with the following input and output parameters
 - in **tt:MotionRegionDetector** - rule type
 - out *analyticsConfListWithSupportingOfMotionRegionDetector* - a list of Analytics configurations
4. If *analyticsConfListWithSupportingOfMotionRegionDetector* is empty, FAIL the test and skip other steps.
5. For each Analytics Configuration *analyticsConf* in *analyticsConfListWithSupportingOfMotionRegionDetector* repeat the following steps:
 - 5.1. ONVIF Client invokes **GetRuleOptions** request with parameters
 - RuleType := `tt:MotionRegionDetector`
 - ConfigurationToken := *analyticsConf.@token*
 - 5.2. DUT responds with **GetRuleOptionsResponse** message with parameters
 - RuleOptions list =: *ruleOptionsList*
 - 5.3. If *ruleOptionsList* does not contain RuleOption with @Name = **MotionRegion** and with @Type = **axt:MotionRegionConfigOptions** (if @Type is present), FAIL the test and skip other steps.
 - 5.4. If RuleOption element with @Name = **MotionRegion** and with @Type = **axt:MotionRegionConfigOptions** (if @Type is present) does not contain **MotionRegionConfigOptions** element, FAIL the test and skip other steps.

Test Result:**PASS –**

- DUT passes all assertions.

FAIL –

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

5.1.3 CREATE MOTION REGION DETECTOR RULE

Test Case ID: ANALYTICS-1-1-3

Specification Coverage: Create Rules (ONVIF Analytics Service Spec)

Feature Under Test: Create Rules

WSDL Reference: analytics.wsdl, media2.wsdl

Test Purpose: To verify adding of Motion Region Detector Rule to an AnalyticsConfiguration by CreateRules operation.

Pre-Requisite: Analytics Service is received from the DUT. Media2 Service is received from the DUT. Rule Engine is supported by the DUT as indicated by the RuleSupport capabilities. Motion Region Detector Rule is supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client retrieves a list of Analytics Configurations that supports tt:MotionRegionDetector type by following the procedure mentioned in [Annex A.2](#) with the following input and output parameters
 - in **tt:MotionRegionDetector** - rule type
 - out *analyticsConfListWithSupportingOfMotionRegionDetector* - a list of Analytics configurations
4. If *analyticsConfListWithSupportingOfMotionRegionDetector* is empty, FAIL the test and skip other steps.
5. ONVIF Client configures media profile with Analytics Configuration from *analyticsConfListWithSupportingOfMotionRegionDetector* list by following the procedure mentioned in [Annex A.4](#) with the following input and output parameters
 - in *analyticsConfListWithSupportingOfMotionRegionDetector* - list of Analytics configurations.
 - out *profile* - media profile.
6. ONVIF Client retrieves Rule Options of tt:MotionRegionDetector type by following the procedure mentioned in [Annex A.3](#) with the following input and output parameters
 - in **tt:MotionRegionDetector** - Rule type
 - in *profile.Configurations.Analytics* - Analytics Configuration

- out *ruleOptions* - Rule Options
7. If *ruleOptionsList* does not contain RuleOption with @Name = **MotionRegion** and with @Type = **axt:MotionRegionConfigOptions** (if @Type is present), FAIL the test and skip other steps.
 8. If RuleOption element with @Name = **MotionRegion** and @Type = **axt:MotionRegionConfigOptions** (if @Type is present) does not contain **MotionRegionConfigOptions** element, FAIL the test and skip other steps.
 9. Set *motionRegionConfigOptions* := RuleOption[0].MotionRegionConfigOptions, where RuleOption[0] is element with @Name = **MotionRegion** and with @Type = **axt:MotionRegionConfigOptions** (if @Type is present).
 10. ONVIF Client calculates free space for adding of new rule with tt:MotionRegionDetector type by following the procedure mentioned in [Annex A.8](#) with the following input and output parameters
 - in **tt:MotionRegionDetector** - rule type
 - in *profile.Configurations.Analytics.token* - a token of Analytics Configuration
 - out *maxInstances* - flag if maxInstances is supported.
 - out *amountOfAdditionalRules* - amount of additional rules.
 11. If *amountOfAdditionalRules* > 0 or *maxInstances*=false, go to step [13 \[26\]](#).
 12. ONVIF Client deletes rule with tt:MotionRegionDetector type by following the procedure mentioned in [Annex A.9](#) with the following input and output parameters
 - in **tt:MotionRegionDetector** - rule type
 - in *profile.Configurations.Analytics.token* - a token of Analytics Configuration
 13. ONVIF Client invokes **CreateRules** request with parameters
 - ConfigurationToken := *profile.Configurations.Analytics.@token*
 - Rule[0].@Name := TestMotionRegion
 - Rule[0].@Type := tt:MotionRegionDetector
 - Rule[0].Parameters.ElementItem[0].@Name := "MotionRegion"
 - Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[0].@x := *profile.Configurations.VideoSource.Bounds.@x*

- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[0].@y :=
profile.Configurations.VideoSource.Bounds.@y
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[1].@x :=
profile.Configurations.VideoSource.Bounds.@x
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[1].@y :=
profile.Configurations.VideoSource.Bounds.@y +
profile.Configurations.VideoSource.Bounds.@height - 1
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[2].@x :=
profile.Configurations.VideoSource.Bounds.@x +
profile.Configurations.VideoSource.Bounds.@width - 1
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[2].@y :=
profile.Configurations.VideoSource.Bounds.@y +
profile.Configurations.VideoSource.Bounds.@height - 1
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[3].@x :=
profile.Configurations.VideoSource.Bounds.@x +
profile.Configurations.VideoSource.Bounds.@width - 1
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[3].@y :=
profile.Configurations.VideoSource.Bounds.@y
- Rule[0].Parameters.ElementItem[0].MotionRegion.@Armed := true if
motionRegionConfigOptions.DisarmSupport = true, otherwise skipped.
- Rule[0].Parameters.ElementItem[0].MotionRegion.@Sensitivity := 1

14. The DUT responds with **CreateRulesResponse** or with SOAP fault response.

15. If DUT responded with SOAP fault response:

15.1. If *maxInstances* = true, fail the test and skip other steps.

15.2. ONVIF Client deletes rule with *tt:MotionRegionDetector* type by following the procedure mentioned in [Annex A.9](#) with the following input and output parameters

- in **tt:MotionRegionDetector** - rule type
- in *profile.Configurations.Analytics.token* - a token of Analytics Configuration

15.3. ONVIF Client invokes **CreateRules** request with parameters

- ConfigurationToken := *profile.Configurations.Analytics.@token*

- Rule[0].@Name := TestMotionRegion
- Rule[0].@Type := tt:MotionRegionDetector
- Rule[0].Parameters.ElementItem[0].@Name := "MotionRegion"
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[0].@x :=
profile.Configurations.VideoSource.Bounds.@x
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[0].@y :=
profile.Configurations.VideoSource.Bounds.@y
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[1].@x :=
profile.Configurations.VideoSource.Bounds.@x
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[1].@y :=
profile.Configurations.VideoSource.Bounds.@y
profile.Configurations.VideoSource.Bounds.@height - 1 +
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[2].@x :=
profile.Configurations.VideoSource.Bounds.@x
profile.Configurations.VideoSource.Bounds.@width - 1 +
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[2].@y :=
profile.Configurations.VideoSource.Bounds.@y
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[3].@x :=
profile.Configurations.VideoSource.Bounds.@x
profile.Configurations.VideoSource.Bounds.@width - 1 +
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[3].@y :=
profile.Configurations.VideoSource.Bounds.@y
profile.Configurations.VideoSource.Bounds.@height - 1 +
- Rule[0].Parameters.ElementItem[0].MotionRegion.@Armed := true if
motionRegionConfigOptions.DisarmSupport = true, otherwise skipped.
- Rule[0].Parameters.ElementItem[0].MotionRegion.@Sensitivity := 1

15.4. The DUT responds with **CreateRulesResponse**.

16. ONVIF Client retrieves updated Rule list by following the procedure mentioned in [Annex A.5](#) with the following input and output parameters

- in *profile.Configurations.Analytics.@token* - Analytics configuration token

- out *updatedRuleList* - Rule list.
17. If *updatedRuleList* does not contain Rule with @Name = TestMotionRegion and with Type = tt:MotionRegionDetector, FAIL the test and skip other steps.
 18. Set *rule* := Rule with @Name = TestMotionRegion and with Type = tt:MotionRegionDetector from *updatedRuleList*.
 19. If *rule*.Parameters does not contain ElementItem with @Name = "MotionRegion", FAIL the test and skip other steps.
 20. If *rule*.Parameters.ElementItem with @Name = "MotionRegion" is not equal to Parameters.ElementItem[0] element from step 13 [26], FAIL the test and skip other steps.
 21. ONVIF Client invokes **DeleteRules** request with parameters
 - ConfigurationToken := *profile*.Configurations.Analytics.@token
 - RuleName := TestMotionRegion
 22. The DUT responds with **DeleteRulesResponse**.
 23. ONVIF Client retrieves updated Rule list by following the procedure mentioned in [Annex A.5](#) with the following input and output parameters
 - in *profile*.Configurations.Analytics.@token - Analytics configuration token
 - out *updatedRuleList* - Rule list.
 24. If *updatedRuleList* contains Rule with @Name = TestMotionRegion and with Type = tt:MotionRegionDetector, FAIL the test and skip other steps.
 25. ONVIF Client restores rule if it was deleted at step 12 [26] and at step 15.2 [27].
 26. ONVIF Client restores media profile if it was changed at step 5 [25].

Test Result:**PASS –**

- DUT passes all assertions.

FAIL –

- DUT did not send **CreateRulesResponse** message.
- DUT did not send **DeleteRules** message.

Note: The following fields are compared at step 20 [29]:

- MotionRegion.Polygon.Point[0].@x
- MotionRegion.Polygon.Point[0].@y
- MotionRegion.Polygon.Point[1].@x
- MotionRegion.Polygon.Point[1].@y
- MotionRegion.Polygon.Point[2].@x
- MotionRegion.Polygon.Point[2].@y
- MotionRegion.Polygon.Point[3].@x
- MotionRegion.Polygon.Point[3].@y
- If **CreateRules** request contained Armed filed:
 - MotionRegion.@Armed
 - MotionRegion.@Sensitivity

5.1.4 MODIFY MOTION REGION DETECTOR RULE

Test Case ID: ANALYTICS-1-1-4

Specification Coverage: Get Rule Options (ONVIF Analytics Service Spec), Modify Rules (ONVIF Analytics Service Spec)

Feature Under Test: Modify Rules

WSDL Reference: analytics.wSDL, media2.wSDL

Test Purpose: To verify modifying of Motion Region Detector Rule by ModifyRules operation.

Pre-Requisite: Analytics Service is received from the DUT. Media2 Service is received from the DUT. Rule Engine is supported by the Device as indicated by the RuleSupport capability. Rule Options is supported by the Device as indicated by the RuleOptionsSupported capability. Motion Region Detector Rule is supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

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

3. ONVIF Client retrieves a list of Analytics Configurations that supports `tt:MotionRegionDetector` type by following the procedure mentioned in [Annex A.2](#) with the following input and output parameters
 - in **tt:MotionRegionDetector** - rule type
 - out *analyticsConfListWithSupportingOfMotionRegionDetector* - a list of Analytics configurations
4. If *analyticsConfListWithSupportingOfMotionRegionDetector* is empty, FAIL the test and skip other steps.
5. ONVIF Client configure media profile with Analytics Configuration from *analyticsConfListWithSupportingOfMotionRegionDetector* list by following the procedure mentioned in [Annex A.4](#) with the following input and output parameters
 - in *analyticsConfListWithSupportingOfMotionRegionDetector* - list of Analytics configurations.
 - out *profile* - media profile.
6. ONVIF Client retrieves Rule Options of `tt:MotionRegionDetector` type by following the procedure mentioned in [Annex A.3](#) with the following input and output parameters
 - in **tt:MotionRegionDetector** - Rule type
 - in *profile.Configurations.Analytics* - Analytics Configuration
 - out *ruleOptions* - Rule Options
7. If *ruleOptionsList* does not contain RuleOption with `@Name = MotionRegion` and with `@Type = axt:MotionRegionConfigOptions` (if `@Type` is present), FAIL the test and skip other steps.
8. If RuleOption element with `@Name = MotionRegion` and `@Type = axt:MotionRegionConfigOptions` (if `@Type` is present) does not contain **MotionRegionConfigOptions** element, FAIL the test and skip other steps.
9. Set *motionRegionConfigOptions* := RuleOption[0].MotionRegionConfigOptions, where RuleOption[0] is element with `@Name = MotionRegion` and with `@Type = axt:MotionRegionConfigOptions` (if `@Type` is present).
10. ONVIF Client creates Motion Region Detector Rule by following the procedure mentioned in [Annex A.10](#) with the following input parameter
 - in *profile* - media profile.

- in *motionRegionConfigOptions* - motion region configuration option.

11. ONVIF Client invokes **ModifyRules** request with parameters

- ConfigurationToken := *profile.Configurations.Analytics.@token*
- Rule[0].@Name := TestMotionRegion
- Rule[0].@Type := tt:MotionRegionDetector
- Rule[0].Parameters.ElementItem[0].@Name := "MotionRegion"
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[0].@x := *profile.Configurations.VideoSource.Bounds.@x*
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[0].@y := *profile.Configurations.VideoSource.Bounds.@y*
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[1].@x := *profile.Configurations.VideoSource.Bounds.@x*
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[1].@y := $[(\textit{profile.Configurations.VideoSource.Bounds.@y} + \textit{profile.Configurations.VideoSource.Bounds.@height} - 1)/2]$
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[2].@x := $[(\textit{profile.Configurations.VideoSource.Bounds.@x} + \textit{profile.Configurations.VideoSource.Bounds.@width} - 1)/2]$
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[2].@y := $[(\textit{profile.Configurations.VideoSource.Bounds.@y} + \textit{profile.Configurations.VideoSource.Bounds.@height} - 1)/2]$
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[3].@x := $[(\textit{profile.Configurations.VideoSource.Bounds.@x} + \textit{profile.Configurations.VideoSource.Bounds.@width} - 1)/2]$
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[3].@y := $[(\textit{profile.Configurations.VideoSource.Bounds.@y} + \textit{profile.Configurations.VideoSource.Bounds.@height} - 1)/2]$
- Rule[0].Parameters.ElementItem[0].MotionRegion.@Armed := false if *motionRegionConfigOptions.DisarmSupport* = true, otherwise skipped.
- Rule[0].Parameters.ElementItem[0].MotionRegion.@Sensitivity := 0

12. The DUT responds with **ModifyRulesResponse**.

13. ONVIF Client retrieves updated Rule list by following the procedure mentioned in [Annex A.5](#) with the following input and output parameters
 - in *profile.Configurations.Analytics.@token* - Analytics configuration token
 - out *updatedRuleList* - Rule list.
14. If *updatedRuleList* does not contain Rule with @Name = "**TestMotionRegion**" and with Type = **tt:MotionRegionDetector**, FAIL the test and skip other steps.
15. Set *rule* := Rule with @Name = "TestMotionRegion" and with Type = **tt:MotionRegionDetector** from *updatedRuleList*.
16. If *rule.Parameters* does not contain ElementItem with @Name = "MotionRegion", FAIL the test and skip other steps.
17. If *rule.Parameters.ElementItem* with @Name = "**MotionRegion**" is not equal to *Parameters.ElementItem[0]* element from step 11 [32], FAIL the test and skip other steps.
18. ONVIF Client invokes **DeleteRules** request with parameters
 - ConfigurationToken := *profile.Configurations.Analytics.@token*
 - RuleName := TestMotionRegion
19. The DUT responds with **DeleteRulesResponse**.
20. ONVIF Client restores rule if it was deleted at step 10 [31].
21. ONVIF Client restores media profile if it was changed at step 5 [31].

Test Result:**PASS –**

- DUT passes all assertions.

FAIL –

- DUT did not send **CreateRulesResponse** message.
- DUT did not send **ModifyRulesResponse** message.
- DUT did not send **DeleteRules** message.

Note: Symbol [] at step 10 [31] means integer part of value (floor function).

Note: The following fields are compared at step 17 [33]:

- MotionRegion.Polygon.Point[0].@x

- MotionRegion.Polygon.Point[0].@y
- MotionRegion.Polygon.Point[1].@x
- MotionRegion.Polygon.Point[1].@y
- MotionRegion.Polygon.Point[2].@x
- MotionRegion.Polygon.Point[2].@y
- MotionRegion.Polygon.Point[3].@x
- MotionRegion.Polygon.Point[3].@y
- If **ModifyRules** request contained Armed filed:
 - MotionRegion.@Armed
 - MotionRegion.@Sensitivity

5.2 Events

5.2.1 MOTION REGION DETECTOR EVENT

Test Case ID: ANALYTICS-2-1-1

Specification Coverage: Motion Region Detector (ONVIF Analytics Service Spec)

Feature Under Test: tns1:RuleEngine/MotionRegionDetector/Motion

WSDL Reference: analytics.wSDL, media2.wSDL

Test Purpose: To verify tns1:RuleEngine/MotionRegionDetector/Motion event format. To verify event generation for tns1:RuleEngine/MotionRegionDetector/Motion.

Pre-Requisite: Analytics Service is received from the DUT. Media2 Service is received from the DUT. Rule Engine is supported by the Device as indicated by the RuleSupport capability. Rule Options is supported by the Device as indicated by the RuleOptionsSupported capability. Motion Region Detector Rule is supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

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

3. ONVIF Client retrieves a list of Analytics Configurations that supports `tt:MotionRegionDetector` type by following the procedure mentioned in [Annex A.2](#) with the following input and output parameters
 - in **tt:MotionRegionDetector** - rule type
 - out *analyticsConfListWithSupportingOfMotionRegionDetector* - a list of Analytics configurations
4. If *analyticsConfListWithSupportingOfMotionRegionDetector* is empty, FAIL the test and skip other steps.
5. ONVIF Client configure media profile with Analytics Configuration from *analyticsConfListWithSupportingOfMotionRegionDetector* list by following the procedure mentioned in [Annex A.4](#) with the following input and output parameters
 - in *analyticsConfListWithSupportingOfMotionRegionDetector* - list of Analytics configurations.
 - out *profile* - media profile.
6. ONVIF Client retrieves Rule Options of `tt:MotionRegionDetector` type by following the procedure mentioned in [Annex A.3](#) with the following input and output parameters
 - in **tt:MotionRegionDetector** - Rule type
 - in *profile.Configurations.Analytics* - Analytics Configuration
 - out *ruleOptions* - Rule Options
7. If *ruleOptionsList* does not contain RuleOption with `@Name = MotionRegion` and with `@Type = axt:MotionRegionConfigOptions` (if `@Type` is present), FAIL the test and skip other steps.
8. If RuleOption element with `@Name = MotionRegion` and `@Type = axt:MotionRegionConfigOptions` (if `@Type` is present) does not contain **MotionRegionConfigOptions** element, FAIL the test and skip other steps.
9. Set *motionRegionConfigOptions* := RuleOption[0].MotionRegionConfigOptions, where RuleOption[0] is element with `@Name = MotionRegion` and with `@Type = axt:MotionRegionConfigOptions` (if `@Type` is present).
10. ONVIF Client creates Motion Region Detector Rule by following the procedure mentioned in [Annex A.10](#) with the following input parameter
 - in *profile* - media profile.

- in *motionRegionConfigOptions* - motion region configuration option.
11. ONVIF Client invokes **GetEventProperties**.
 12. The DUT responds with **GetEventPropertiesResponse** with parameters
 - TopicNamespaceLocation list
 - FixedTopicSet
 - TopicSet =: *topicSet*
 - TopicExpressionDialect list
 - MessageContentFilterDialect list
 - MessageContentSchemaLocation list
 13. If *topicSet* does not contain **tns1:RuleEngine/MotionRegionDetector/Motion** topic, FAIL the test and skip other steps.
 14. Set *topic* := **tns1:RuleEngine/MotionRegionDetector/Motion** topic from *topicSet*.
 15. If *topic*.MessageDescription.IsProperty is not equal to true, FAIL the test and skip other steps.
 16. If *topic* does not contain MessageDescription.Source.SimpleItemDescription item with Name = "VideoSource", FAIL the test and skip other steps.
 17. If *topic*.MessageDescription.Source.SimpleItemDescription with Name = "VideoSource" does not have Type = "tt:ReferenceToken", FAIL the test and skip other steps.
 18. If *motionRegionConfigOptions*.MotionRegionConfigOptions.RuleNotification = true:
 - 18.1. If *topic* does not contain MessageDescription.Source.SimpleItemDescription item with Name = "RuleName", FAIL the test and skip other steps.
 - 18.2. If *topic*.MessageDescription.Source.SimpleItemDescription with Name = "RuleName" does not have Type = "xs:string", FAIL the test and skip other steps.
 19. If *topic* does not contain MessageDescription.Data.SimpleItemDescription item with Name = "State", FAIL the test and skip other steps.
 20. If *topic*.MessageDescription.Data.SimpleItemDescription with Name = "State" does not have Type = "xs:boolean", FAIL the test and skip other steps.
 21. ONVIF Client creates PullPoint subscription for the specified topic by following the procedure mentioned in [Annex A.6](#) with the following input and output parameters

- in **"tns1:RuleEngine/MotionRegionDetector/Motion"** - Notification Topic
- out *s* - Subscription reference
- out *currentTime* - current time for the DUT
- out *terminationTime* - Subscription termination time

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

22.1. ONVIF Client waits for time $t := \min\{(tt-ct)/2, 1 \text{ second}\}$.

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

- Timeout := PT60S
- MessageLimit := 1

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

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

22.4. If *notificationMessageList* contains more than one notification, FAIL the test and skip other steps.

22.5. If *notificationMessageList* is not empty and *notificationMessageList*[0].Topic is not equal to **"tns1:RuleEngine/MotionRegionDetector/Motion"**, FAIL the test and skip other steps.

22.6. If *notificationMessageList* is not empty and *notificationMessageList*[0].PropertyOperation = "Initialized" and *notificationMessageList*[0] has Source.SimpleItem with Name = "VideoSource" and with Value = *profile.Configurations.VideoSource.SourceToken*:

22.6.1. If *motionRegionConfigOptions.MotionRegionConfigOptions.RuleNotification* is not equal to true, go to step 23 [38].

22.6.2. If *notificationMessageList*[0] has Source.SimpleItem with Name = "RuleName" and with Value = "TestMotionRegion", go to step 23 [38].

22.7. If *timeout1* timeout expires for step 22 without Notification corresponds to step 22.6 [37], FAIL the test and skip other steps.

23. If *notificationMessageList*[0] does not have *Data.SimpleItem* with Name = "State" and with Value with type = "xs:boolean", FAIL the test and skip other steps.
24. ONVIF Client invokes **DeleteRules** request with parameters
- ConfigurationToken := *profile.Configurations.Analytics.@token*
 - RuleName := TestMotionRegion
25. The DUT responds with **DeleteRulesResponse**.
26. ONVIF Client restores rule if it was deleted at step 10 [35].
27. ONVIF Client restores media profile if it was changed at step 5 [35].
28. ONVIF Client deletes PullPoint subscription by following the procedure mentioned in [Annex A.7](#) with the following input and output parameters
- in s - Subscription reference

Test Result:**PASS –**

- DUT passes all assertions.

FAIL –

- DUT did not send **CreateRulesResponse** message.
- DUT did not send **DeleteRules** message.
- DUT did not send **GetEventPropertiesResponse** message.
- DUT did not send **PullMessagesResponse** message.

5.3 Capabilities

5.3.1 Get Services and Get Analytics Service Capabilities Consistency

Test Case ID: ANALYTICS-3-1-1

Specification Coverage: Capability exchange (ONVIF Core Specification), Capabilities (ONVIF Analytics Service Spec)

Feature under test: GetServices, GetServiceCapabilities (Analytics)

WSDL Reference: devicemgmt.wsdl, analytics.wsdl

Test Purpose: To verify getting Analytics Service using GetServices request. To verify Get Services and Analytics Service Capabilities consistency.

Pre-Requisite: Analytics 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 invokes **GetServices** message with parameters:
 - IncludeCapability := false
4. The DUT responds with a **GetServicesResponse** message with parameters:
 - Service list =: *listOfServicesWithoutCapabilities*
5. If *listOfServicesWithoutCapabilities* does not contain item with Namespace = "http://www.onvif.org/ver20/analytics/wsdl", FAIL the test and skip other steps.
6. Set *analyticsServ* := item from *listOfServicesWithoutCapabilities* list with Namespace = "http://www.onvif.org/ver20/analytics/wsdl".
7. If *analyticsServ.Capabilities* is specified, FAIL the test and skip other steps.
8. ONVIF Client invokes **GetServices** message with parameters:
 - IncludeCapability := true
9. The DUT responds with a **GetServicesResponse** message with parameters:
 - Service list =: *listOfServicesWithCapabilities*
10. If *listOfServicesWithCapabilities* does not contain item with Namespace = "http://www.onvif.org/ver20/analytics/wsdl", FAIL the test and skip other steps.
11. Set *analyticsServ* := item from *listOfServicesWithCapabilities* list with Namespace = "http://www.onvif.org/ver20/analytics/wsdl".
12. If *analyticsServ.Capabilities* is not specified, FAIL the test and skip other steps.
13. If *analyticsServ.Capabilities* does not contain valid Capabilities element for Analytics service from "http://www.onvif.org/ver20/analytics/wsdl" namespace, FAIL the test and skip other steps.

14. ONVIF Client invokes **GetServiceCapabilities** (Analytics) request.
15. The DUT responds with **GetServiceCapabilitiesResponse** message with parameters
 - Capabilities =: *cap*
16. If *cap* differs from *analyticsServ.Capabilities.Capabilities* (see Note at the end of the test), FAIL the test.

Test Result:**PASS –**

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetServicesResponse** messages.
- The DUT did not send **GetServiceCapabilitiesResponse** message.

Note: The following fields are compared at step 16:

- RuleSupport
- AnalyticsModuleSupport
- RuleOptionsSupported
- AnalyticsModuleOptionsSupported
- SupportedMetadata

5.3.2 Analytics Service Capabilities

Test Case ID: ANALYTICS-3-1-2

Specification Coverage: Capabilities (ONVIF Analytics Service Spec)

Feature under test: GetServiceCapabilities (Analytics Service)

WSDL Reference: analytics.wsdl

Test Purpose: To verify Analytics Service Capabilities.

Pre-Requisite: Analytics 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 invokes **GetServiceCapabilities** request.
4. The DUT responds with **GetServiceCapabilitiesResponse** message with parameters
 - Capabilities =: *cap*
5. If *cap.SupportedMetadata* = true and *cap.AnalyticsModuleSupport* = false or skipped, FAIL the test.

Test Result:**PASS –**

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetServicesResponse** messages.
- The DUT did not send **GetServiceCapabilitiesResponse** message.

5.4 Analytics Modules

5.4.1 Get Supported Analytics Modules

Test Case ID: ANALYTICS-4-1-1

Specification Coverage: GetSupportedAnalyticsModules (ONVIF Analytics Service Spec)

Feature under test: GetSupportedAnalyticsModules (Analytics)

WSDL Reference: analytics.wsdl

Test Purpose: To verify getting supported analytics modules using GetSupportedAnalyticsModules request.

Pre-Requisite: Analytics Service was received from the DUT. Media2 Service was received from the DUT. Analytics Modules is supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.
3. ONVIF Client retrieves a list of Analytics Configurations by following the procedure mentioned in [Annex A.1](#) with the following input and output parameters
 - out *analyticsConfList* - a list of Analytics configurations
4. For each Analytics Configuration *analyticsConf* in *analyticsConfList* repeat the following steps:
 - 4.1. ONVIF Client invokes **GetSupportedAnalyticsModules** request with parameters
 - ConfigurationToken := *analyticsConf.@token*
 - 4.2. DUT responds with **GetSupportedAnalyticsModulesResponse** message with parameters
 - SupportedAnalyticsModules =: *supportedModules*
 - 4.3. For each Analytics Module Description *analyticsModuleDescription* in *supportedModules.AnalyticsModuleDescription* list repeat the following steps:
 - 4.3.1. If *analyticsModuleDescription* does not have *maxInstances* attribute, FAIL the test, restore the DUT state, and skip other steps.
 - 4.3.2. If at least one item in *analyticsModuleDescription.Parameters* list (SimpleItemDescription or ElementItemDescription item) has the same Name value with other item from the same list, FAIL the test, restore the DUT state, and skip other steps.
 - 4.3.3. If at least one item in *analyticsModuleDescription.Messages* list (Source, or Key, or Data; SimpleItemDescription or ElementItemDescription item) has the same Name value with other item from the same list, FAIL the test, restore the DUT state, and skip other steps.
 - 4.3.4. If *analyticsModuleDescription.ParentTopic* is not valid topic (see [Annex A.11](#)), FAIL the test, restore the DUT state, and skip other steps.

Test Result:**PASS –**

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetSupportedAnalyticsModulesResponse** messages.

5.4.2 Get Analytics Module Options

Test Case ID: ANALYTICS-4-1-2

Specification Coverage: GetAnalyticsModuleOptions (ONVIF Analytics Service Spec)

Feature under test: GetAnalyticsModuleOptions (Analytics)

WSDL Reference: analytics.wsdl

Test Purpose: To verify getting supported analytics module options using GetAnalyticsModuleOptions request.

Pre-Requirement: Analytics Service was received from the DUT. Media2 Service was received from the DUT. Analytics Modules is supported by the DUT. Analytics Module Options is supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client retrieves a list of Analytics Configurations by following the procedure mentioned in [Annex A.1](#) with the following input and output parameters
 - out *analyticsConfList* - a list of Analytics configurations
4. For each Analytics Configuration *analyticsConf* in *analyticsConfList* repeat the following steps:
 - 4.1. ONVIF Client invokes **GetSupportedAnalyticsModules** request with parameters
 - ConfigurationToken := *analyticsConf.@token*
 - 4.2. DUT responds with **GetSupportedAnalyticsModulesResponse** message with parameters
 - SupportedAnalyticsModules =: *supportedModules*
 - 4.3. ONVIF Client invokes **GetAnalyticsModuleOptions** request with parameters
 - Type is skipped
 - ConfigurationToken := *analyticsConf.@token*

- 4.4. DUT responds with **GetAnalyticsModuleOptionsResponse** message with parameters
 - Options list =: *analyticsModuleOptionsFullList*
- 4.5. If at least one item in *analyticsModuleOptionsFullList* list contains RuleType, FAIL the test, restore the DUT state, and skip other steps.
- 4.6. For each Analytics Module Description *analyticsModuleDescription* in *supportedModules.AnalyticsModuleDescription* list repeat the following steps:
 - 4.6.1. ONVIF Client invokes **GetAnalyticsModuleOptions** request with parameters
 - Type := *analyticsModuleDescription.Name*
 - ConfigurationToken := *analyticsConf.@token*
 - 4.6.2. DUT responds with **GetAnalyticsModuleOptionsResponse** message with parameters
 - Options list =: *analyticsModuleOptionsList*
 - 4.6.3. If at least one item in *analyticsModuleOptionsList* list contains RuleType, FAIL the test, restore the DUT state, and skip other steps.
 - 4.6.4. If at least one item in *analyticsModuleOptionsList* list contains AnalyticsModule which is not equal to *analyticsModuleDescription.Name*, FAIL the test, restore the DUT state, and skip other steps.
 - 4.6.5. If *analyticsModuleOptionsFullList* list does not contain all items from *analyticsModuleOptionsList* list (AnalyticsModule and Name to be used as unique identifier), FAIL the test, restore the DUT state, and skip other steps.
 - 4.6.6. For each Parameters Description item *parametersDescriptionItem* in *analyticsModuleDescription.Parameters* list (SimpleItemDescription or ElementItemDescription items) list repeat the following steps:
 - 4.6.6.1. If *analyticsModuleOptionsList* list does not contains item with Name = *parametersDescriptionItem.Name*, FAIL the test, restore the DUT state, and skip other steps.

Test Result:**PASS –**

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetSupportedAnalyticsModulesResponse** messages.
- The DUT did not send **GetAnalyticsModuleOptionsResponse** messages.

5.4.3 Get Analytics Modules

Test Case ID: ANALYTICS-4-1-3

Specification Coverage: GetAnalyticsModules (ONVIF Analytics Service Spec)

Feature under test: GetAnalyticsModules (Analytics)

WSDL Reference: analytics.wsdl

Test Purpose: To verify getting supported analytics modules using GetAnalyticsModules request.

Pre-Requirement: Analytics Service was received from the DUT. Media2 Service was received from the DUT. Analytics Modules is supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client retrieves a list of Analytics Configurations by following the procedure mentioned in [Annex A.1](#) with the following input and output parameters
 - out *analyticsConfList* - a list of Analytics configurations
4. For each Analytics Configuration *analyticsConf* in *analyticsConfList* repeat the following steps:
 - 4.1. ONVIF Client invokes **GetSupportedAnalyticsModules** request with parameters
 - ConfigurationToken := *analyticsConf.@token*
 - 4.2. DUT responds with **GetSupportedAnalyticsModulesResponse** message with parameters
 - SupportedAnalyticsModules =: *supportedModules*
 - 4.3. ONVIF Client invokes **GetAnalyticsModules** request with parameters
 - ConfigurationToken := *analyticsConf.@token*

- 4.4. DUT responds with **GetAnalyticsModulesResponse** message with parameters
- AnalyticsModule list =: *analyticsModuleList*
- 4.5. If at least one item in *analyticsModuleList* list contains Type which is not equal to Name field of at least one item at *supportedModules.AnalyticsModuleDescription* list, FAIL the test, restore the DUT state, and skip other steps.
- 4.6. For each Analytics Module Description *analyticsModuleDescription* in *supportedModules.AnalyticsModuleDescription* list repeat the following steps:
- 4.6.1. If *analyticsModuleDescription.fixed* = true:
- 4.6.1.1. If *analyticsModuleList* does not contain item with Type = *analyticsModuleDescription.Name*, FAIL the test, restore the DUT state, and skip other steps.
- 4.6.2. If *analyticsModuleList* contains item with Type = *analyticsModuleDescription.Name*:
- 4.6.2.1. Set *analyticsModule* := *analyticsModuleList*[Type = *analyticsModuleDescription.Name*].
- 4.6.2.2. If *analyticsModule.Parameters.SimpleItem* contains at least one SimpleItem which does not have item with the same Name in *analyticsModuleDescription.Parameters.SimpleItemDescription* list, FAIL the test, restore the DUT state, and skip other steps.
- 4.6.2.3. If *analyticsModule.Parameters.ElementItem* contains at least one ElementItem which does not have item with the same Name in *analyticsModuleDescription.Parameters.ElementItemDescription* list, FAIL the test, restore the DUT state, and skip other steps.
- 4.6.2.4. If *analyticsModuleDescription.Parameters.SimpleItemDescription* contains at least one SimpleItemDescription which does not have item with the same Name in *analyticsModule.Parameters.SimpleItem* list, FAIL the test, restore the DUT state, and skip other steps.
- 4.6.2.5. If *analyticsModuleDescription.Parameters.ElementItemDescription* contains at least one ElementItemDescription which does not have item with the same Name in

analyticsModule.Parameters.ElementItem list, FAIL the test, restore the DUT state, and skip other steps.

Test Result:**PASS –**

- DUT passes all assertions.

FAIL –

- The DUT did not send **GetSupportedAnalyticsModulesResponse** messages.
- The DUT did not send **GetAnalyticsModulesResponse** messages.

5.4.4 Object Classification Metadata

Test Case ID: ANALYTICS-4-1-5

Specification Coverage: Object Class descriptor (ONVIF Analytics Service Spec)

Feature under test: GetSupportedMetadata (Analytics)

WSDL Reference: analytics.wsdl

Test Purpose: To verify Object Class descriptor.

Pre-Requisite: Analytics Service was received from the DUT. Object Classification feature is supported by the DUT.

Test Configuration: ONVIF Client and DUT

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client retrieves a list of supported metadata by following the procedure mentioned in [Annex A.14](#) with the following input and output parameters
 - out *analyticsModuleList* - Analytics Module List with Metadata Info
4. For each Analytics Module *analyticsModuleWithMetadataInfo* from *analyticsModuleList* repeat the following steps:
 - 4.1. For each *Object* from *analyticsModuleWithMetadataInfo*

4.1.1. If *Object* contains *Appearance.Class*

4.1.1.1. Set *LikelihoodSum* := {all *ClassCandidate.Likelihood* + all
Extension.OtherTypes.Likelihood + all *Type.@Likelihood*}

4.1.1.2. If *LikelihoodSum* > 1, ONVIF Client provides warning.

Test Result:

PASS –

- DUT passes all assertions.

FAIL –

- None.

Annex A Helper Procedures and Additional Notes

A.1 Get Analytics Configurations List

Name: HelperGetAnalyticsConfigurationsList

Procedure Purpose: Helper procedure to retrieve Analytics Configurations List.

Pre-requisite: Media2 Service is received from the DUT.

Input: None.

Returns: Analytics Configurations list (*analyticsConfList*).

Procedure:

1. ONVIF Client invokes **GetAnalyticsConfigurations** request with parameters
 - ConfigurationToken skipped
 - ProfileToken skipped
2. The DUT responds with **GetAnalyticsConfigurationsResponse** with parameters
 - Configurations list =: *analyticsConfList*
3. If *analyticsConfList* is empty, FAIL the test.

Procedure Result:

PASS –

- DUT passes all assertions.

FAIL –

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

A.2 Get List of Analytics Configurations With Supporting of Required Rule Type

Name: HelperGetAnalyticsConfigurationSupportsRequiredRuleTypeList

Procedure Purpose: Helper procedure to retrieve full list of Analytics Configuration that supports required rule type.

Pre-requisite: Analytics Service is received from the DUT. Rule Engine is supported by the Device as indicated by the RuleSupport capabilities.

Input: Rule Type (*ruleType*).

Returns: List of Analytics Configuration that supports rule with type equals to *ruleType* (*analyticsConfSupportsRuleTypeList*).

Procedure:

1. ONVIF Client retrieves a list of Analytics Configurations by following the procedure mentioned in [Annex A.1](#) with the following input and output parameters
 - out *analyticsConfList* - a list of Analytics configurations
2. For each Analytics Configuration *analyticsConf* in *analyticsConfList* repeat the following steps:
 - 2.1. ONVIF Client invokes **GetSupportedRules** request with parameters
 - ConfigurationToken := *analyticsConf.@token*
 - 2.2. DUT responds with **GetSupportedRulesResponse** message with parameters
 - SupportedRules =: *supportedRules*
 - 2.3. If *supportedRules* contains RuleDescription element with Name value is equal to *ruleType*, set *analyticsConfSupportsRuleTypeList* := *analyticsConfSupportsRuleTypeList* + *analyticsConf.@token*

Procedure Result:

PASS –

- DUT passes all assertions.

FAIL –

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

A.3 Get Specific Rule Options

Name: HelperGetSpecificRuleOptions

Procedure Purpose: Helper procedure to retrieve options of required rule type.

Pre-requisite: Analytics Service is received from the DUT. Rule Engine is supported by the Device as indicated by the RuleSupport capabilities.

Input: Analytics Configuration (*analyticsConf*), Rule Type (*ruleType*).

Returns: Rule Options *ruleOptions* of *ruleType*.

Procedure:

1. ONVIF Client invokes **GetRuleOptions** request with parameters
 - RuleType := *ruleType*
 - ConfigurationToken := *analyticsConf.@token*
2. DUT responds with **GetRuleOptionsResponse** message with parameters
 - RuleOptions list =: *ruleOptionsList*
3. If **ruleOptionsList** contains more than one RuleOptions element, FAIL the test and skip other steps.
4. If **ruleOptionsList** is empty, FAIL the test and skip other steps.
5. Set *ruleOptions* := **ruleOptionsList**[0].

Procedure Result:

PASS –

- DUT passes all assertions.

FAIL –

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

A.4 Configure Media Profile with required Analytics Configuration

Name: HelperConfigureMediaProfileWithRequiredAnalytics

Procedure Purpose: Helper procedure to configure Media Profile to contain required Analytics Configuration.

Pre-requisite: Media2 Service is received from the DUT. Analytics is supported by the DUT.

Input: List of Analytics configurations *analyticsConfList*

Returns: Media Profile (*profile*) that contains Analytics Configuration from *analyticsConfList* and Video Source Configuration.

Procedure:

1. ONVIF Client invokes **GetProfiles** request with parameters
 - Token skipped
 - Type[0] := VideoSource
 - Type[1] := Analytics
2. The DUT responds with **GetProfilesResponse** message with parameters
 - Profiles list =: *profileList*
3. For each Media Profile *profile1* in *profileList* with both Configuration.VideoSource and Configuration.Analytics repeat the following steps:
 - 3.1. For each Analytics (*analytics*) in *analyticsConfList*:
 - 3.1.1. If *profile1*.Configuration.Analytics.@token value is equal to *analytics*.@token, set *profile* := *profile1* and skip other steps in procedure.
4. For each Media Profile *profile1* in *profileList* that contains VideoSource configuration repeat the following steps:
 - 4.1. ONVIF Client invokes **GetAnalyticsConfigurations** request with parameters
 - ConfigurationToken skipped
 - ProfileToken := *profile*.@token
 - 4.2. The DUT responds with **GetAnalyticsConfigurationsResponse** message with parameters
 - Configurations list =: *acList*
 - 4.3. If *acList* contains analytics (*analytics*) from *analyticsConfList* (comparing by analytics token):
 - 4.3.1. ONVIF Client invokes **AddConfiguration** request with parameters
 - ProfileToken := *profile1*.@token
 - Name skipped
 - Configuration[0].Type := Analytics
 - Configuration[0].Token := *analytics*.Configurations.@token

4.3.2. The DUT responds with **AddConfigurationResponse** message.

4.3.3. Set *profile* := *profile1* and skip other steps in procedure.

5. FAIL the test and skip other steps.

Procedure Result:

PASS –

- DUT passes all assertions.

FAIL –

- DUT did not send **GetProfilesResponse** message.
- DUT did not send **GetAnalyticsConfigurationsResponse** message.
- DUT did not send **AddConfigurationResponse** message.

A.5 Get Rules

Name: HelperGetRules

Procedure Purpose: Helper procedure to retrieve Rules list for Analytics configuration.

Pre-requisite: Media2 Service is received from the DUT. Rule Engine is supported by the Device as indicated by the RuleSupport capabilities.

Input: Analytics configuration token (*analyticsToken*).

Returns: Rule list (*ruleList*).

Procedure:

1. ONVIF Client invokes **GetRules** request with parameters
 - ConfigurationToken := (*analyticsToken*)
2. The DUT responds with **GetRulesResponse** with parameters
 - Rule list =: *ruleList*

Procedure Result:

PASS –

- DUT passes all assertions.

FAIL –

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

A.6 Create Pull Point Subscription

Name: HelperCreatePullPointSubscription

Procedure Purpose: Helper procedure to create PullPoint Subscription with specified Topic.

Pre-requisite: Event Service is received from the DUT.

Input: Notification Topic (*topic*).

Returns: Subscription reference (*s*), current time for the DUT (*ct*), subscription termination time (*tt*).

Procedure:

1. ONVIF Client invokes **CreatePullPointSubscription** request with parameters
 - Filter.TopicExpression := *topic*
 - Filter.TopicExpression.@Dialect := "http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet"
2. The DUT responds with **CreatePullPointSubscriptionResponse** message with parameters
 - SubscriptionReference =: *s*
 - CurrentTime =: *ct*
 - TerminationTime =: *tt*

Procedure Result:**PASS –**

- DUT passes all assertions.

FAIL –

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

A.7 Delete Subscription

Name: HelperDeleteSubscription

Procedure Purpose: Helper procedure to delete subscription.

Pre-requisite: Event Service is received from the DUT.

Input: Subscription reference (s)

Returns: None

Procedure:

1. ONVIF Client sends an **Unsubscribe** to the subscription endpoint s.
2. The DUT responds with **UnsubscribeResponse** message.

Procedure Result:

PASS –

- DUT passes all assertions.

FAIL –

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

A.8 Calculate Free Space for Rule

Name: Annex_HelperCalculateFreeSpaceForRule

Procedure Purpose: Helper procedure to calculate free space for additional rules with required Rule Type for requested Analytics Configuration.

Pre-requisite: Analytics Service is received from the DUT. Rule Engine is supported by the Device as indicated by the RuleSupport capabilities.

Input: Analytics Configuration token *analyticsConfigToken*. Required Rule Type *ruleType*.

Returns: Flag if maxInstances is supported (*maxInstances*). Amount of additional rules with requested type that may be added for requested Analytics Configuration (*amountOfAdditionalRules*) (optional, returned in case *maxInstances* = true).

Procedure:

1. ONVIF Client invokes **GetSupportedRules** request with parameters
 - ConfigurationToken := *analyticsConfigToken*
2. DUT responds with **GetSupportedRulesResponse** message with parameters

- SupportedRules =: *supportedRules*
3. If *supportedRules* does not contain RuleDescription element with Name value is equal to *ruleType*, FAIL the test and skip other steps.
 4. Set *rule* := *supportedRules*.RuleDescription[0], where RuleDescription[0] is RuleDescription element with Name value is equal to *ruleType*.
 5. If *rule* does not contain maxInstances attribute, set *maxInstances* := false, return it in test procedure and skip other annex steps.
 6. Set *maxInstances* := true.
 7. ONVIF Client invokes **GetRules** request with parameters
 - ConfigurationToken := *analyticsConfigToken*
 8. DUT responds with **GetRulesResponse** message with parameters
 - Rule list =: *ruleList*
 9. Set *amountOfExistingRules* := amount of Rules in *ruleList* with Type = *ruleType*.
 10. Set *amountOfAdditionalRules* := *rule*.maxInstances - *amountOfExistingRules*.

Procedure Result:**PASS –**

- DUT passes all assertions.

FAIL –

- DUT did not send **GetSupportedRulesResponse** message.
- DUT did not send **GetRulesResponse** message.

A.9 Delete Rule with Requested Type

Name: HelperDeleteRuleWithRequestedType

Procedure Purpose: Helper procedure to delete existing Rule with requested type for specified Analytics Configuration.

Pre-requisite: Media2 Service is received from the DUT. Rule Engine is supported by the Device as indicated by the RuleSupport capabilities.

Input: Analytics configuration token (*analyticsToken*). Rule type to delete (*ruleType*).

Returns: None.

Procedure:

1. ONVIF Client invokes **GetRules** request with parameters
 - ConfigurationToken := *analyticsToken*
2. DUT responds with **GetRulesResponse** message with parameters
 - Rule list =: *ruleList*
3. Set *ruleToDelete* := *ruleList*[0], where *ruleList*[0] is the first Rule with Type = *ruleType*.
4. ONVIF Client invokes **DeleteRules** request with parameters
 - ConfigurationToken := (*analyticsToken*)
 - RuleName[0] := *ruleToDelete*.Name
5. The DUT responds with **DeleteRulesResponse**.

Procedure Result:

PASS –

- DUT passes all assertions.

FAIL –

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

A.10 Create Motion Region Detector Rule

Name: HelperCreateMotionRegionDetectorRule

Procedure Purpose: Helper procedure to create Motion Region Detector Rule.

Pre-requisite: Media2 Service is received from the DUT. Rule Engine is supported by the Device as indicated by the RuleSupport capabilities.

Input: Media Profile (*profile*). Rule Options *ruleOptions*.

Returns: None.

Procedure:

1. ONVIF Client calculates free space for adding of new rule with `tt:MotionRegionDetector` type by following the procedure mentioned in [Annex A.8](#) with the following input and output parameters
 - in **tt:MotionRegionDetector** - rule type
 - in `profile.Configurations.Analytics.token` - a token of Analytics Configuration
 - out `maxInstances` - flag if `maxInstances` is supported.
 - out `amountOfAdditionalRules` - amount of additional rules.
2. If `amountOfAdditionalRules > 0` or `maxInstances=false`, go to step 4 [\[58\]](#).
3. ONVIF Client deletes rule with `tt:MotionRegionDetector` type by following the procedure mentioned in [Annex A.9](#) with the following input and output parameters
 - in **tt:MotionRegionDetector** - rule type
 - in `profile.Configurations.Analytics.token` - a token of Analytics Configuration
4. ONVIF Client invokes **CreateRules** request with parameters
 - ConfigurationToken := `profile.Configurations.Analytics.@token`
 - Rule[0].@Name := TestMotionRegion
 - Rule[0].@Type := `tt:MotionRegionDetector`
 - Rule[0].Parameters.ElementItem[0].@Name := "MotionRegion"
 - Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[0].@x := `profile.Configurations.VideoSource.Bounds.@x`
 - Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[0].@y := `profile.Configurations.VideoSource.Bounds.@y`
 - Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[1].@x := `profile.Configurations.VideoSource.Bounds.@x`
 - Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[1].@y := `profile.Configurations.VideoSource.Bounds.@y`
+ `profile.Configurations.VideoSource.Bounds.@height - 1`
 - Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[2].@x := `profile.Configurations.VideoSource.Bounds.@x`
+ `profile.Configurations.VideoSource.Bounds.@width - 1`

- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[2].@y :=
profile.Configurations.VideoSource.Bounds.@y +
profile.Configurations.VideoSource.Bounds.@height - 1
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[3].@x :=
profile.Configurations.VideoSource.Bounds.@x +
profile.Configurations.VideoSource.Bounds.@width - 1
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[3].@y :=
profile.Configurations.VideoSource.Bounds.@y
- Rule[0].Parameters.ElementItem[0].MotionRegion.@Armed := true if
motionRegionConfigOptions.DisarmSupport = true, otherwise skipped.
- Rule[0].Parameters.ElementItem[0].MotionRegion.@Sensitivity := 1

5. The DUT responds with **CreateRulesResponse** or SOAP fault response.

6. If DUT responded with SOAP fault response:

6.1. If *maxInstances* = true, fail the test and skip other steps.

6.2. ONVIF Client deletes rule with *tt:MotionRegionDetector* type by following the procedure mentioned in [Annex A.9](#) with the following input and output parameters

- in **tt:MotionRegionDetector** - rule type
- in *profile.Configurations.Analytics.token* - a token of Analytics Configuration

6.3. ONVIF Client invokes **CreateRules** request with parameters

- ConfigurationToken := *profile.Configurations.Analytics.@token*
- Rule[0].@Name := TestMotionRegion
- Rule[0].@Type := *tt:MotionRegionDetector*
- Rule[0].Parameters.ElementItem[0].@Name := "MotionRegion"
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[0].@x :=
profile.Configurations.VideoSource.Bounds.@x
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[0].@y :=
profile.Configurations.VideoSource.Bounds.@y

- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[1].@x :=
profile.Configurations.VideoSource.Bounds.@x
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[1].@y :=
profile.Configurations.VideoSource.Bounds.@y +
profile.Configurations.VideoSource.Bounds.@height - 1
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[2].@x :=
profile.Configurations.VideoSource.Bounds.@x +
profile.Configurations.VideoSource.Bounds.@width - 1
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[2].@y :=
profile.Configurations.VideoSource.Bounds.@y
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[3].@x :=
profile.Configurations.VideoSource.Bounds.@x +
profile.Configurations.VideoSource.Bounds.@width - 1
- Rule[0].Parameters.ElementItem[0].MotionRegion.Polygon.Point[3].@y :=
profile.Configurations.VideoSource.Bounds.@y +
profile.Configurations.VideoSource.Bounds.@height - 1
- Rule[0].Parameters.ElementItem[0].MotionRegion.@Armed := true if
motionRegionConfigOptions.DisarmSupport = true, otherwise skipped.
- Rule[0].Parameters.ElementItem[0].MotionRegion.@Sensitivity := 1

6.4. The DUT responds with **CreateRulesResponse**.

Procedure Result:

PASS –

- DUT passes all assertions.

FAIL –

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

A.11 Topic Format Verification

Name: HelperTopicCheck

Procedure Purpose: Helper procedure to verifye topic format.

Pre-requisite: None.

Input: Topic to be verified (*topic*).

Returns: None.

Procedure:

1. If *topic* contains at least one whitespace, FAIL the test, restore the DUT state, and skip other steps.
2. If *topic* does not correspond to the format defined in [Annex A.12](#), FAIL the test, restore the DUT state, and skip other steps.
3. If *topic* contains at least one namespace prefix which namespace is not declared according to [XML Schema Part 2], section 3.2.18 QName, FAIL the test, restore the DUT state, and skip other steps.

Procedure Result:

PASS –

- DUT passes all assertions.

FAIL –

- DUT does not pass all assertions.

A.12 Valid Topic Format

Topic shall correspond to the following format in Extended Backus Naur Form:

- TopicExpression ::= TopicPath
- TopicPath ::= RootTopic ChildTopicExpression*
- RootTopic ::= QName
- ChildTopicExpression ::= '/' ChildTopicName
- ChildTopicName ::= QName | NCName

A.13 Get Installed Analytics Modules

Name: HelperGetInstalledAnalyticsModules

Procedure Purpose: Helper procedure to retrieve full list of Analytics Modules currently installed in a DUT.

Pre-requisite: Media2 Service is received from the DUT. Analytics Modules is supported by the DUT.

Input: None.

Returns: Installed Analytics Module List (*installedAnalyticsModuleList*).

Procedure:

1. ONVIF Client retrieves a list of Analytics Configurations by following the procedure mentioned in [Annex A.1](#) with the following input and output parameters
 - out *analyticsConfList* - a list of Analytics configurations
2. For each Analytics Configuration *analyticsConf* in *analyticsConfList* repeat the following steps:
 - 2.1. ONVIF Client invokes **GetAnalyticsModules** request with parameters
 - ConfigurationToken := *analyticsConf.@token*
 - 2.2. DUT responds with **GetAnalyticsModulesResponse** message with parameters
 - AnalyticsModule list =: *analyticsModuleList*
 - 2.3. Set *installedAnalyticsModuleList* := *installedAnalyticsModuleList* + *analyticsModuleList*

Procedure Result:

PASS –

- DUT passes all assertions.

FAIL –

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

A.14 Get Supported Metadata

Name: HelperGetSupportedMetadata

Procedure Purpose: Helper procedure to retrieve full list of metadata supported by a DUT.

Pre-requisite: Object Classification feature is supported by the DUT.

Input: None.

Returns: Analytics Module List with Metadata Info (*analyticsModuleList*).

Procedure:

1. ONVIF Client invokes **GetSupportedMetadata** request with parameters
 - Type skipped
2. DUT responds with **GetSupportedMetadataResponse** message with parameters
 - AnalyticsModule list =: *analyticsModuleList*

Procedure Result:

PASS –

- DUT passes all assertions.

FAIL –

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