ONVIF^{TM}

Access Control Test Specification

Version 13.12

December, 2013

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

Ver.	Date	Description
13.06	Jun, 2013	First issue
13.12	Dec, 2013	Minor updates.

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 needed to verify the [ONVIF Network Interface Specs] and [ONVIF Conformance] requirements. And also 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 Access Control Test Specification acts as a supplementary document to the [ONVIF Network Interface Specs], illustrating necessary test cases to be executed and passed. And 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 Access Control Test Specification defines and regulates the conformance testing procedure for the ONVIF conformant devices. Conformance testing is meant to be functional black-box testing. The objective of this specification is to provide test cases for testing individual requirements of ONVIF devices according to ONVIF Access Control service which is defined in [ONVIF Network Interface Specs].

The principal purposes are:

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

This specification does not address the following:

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

The set of ONVIF Test Specification will not cover the complete set of requirements as defined in [ONVIF Network Interface Specs]; instead it would cover its subset.

This ONVIF Access Control Test Specification covers ONVIF Access Control service and Access Control Events which are a functional block of [ONVIF Network Interface Specs]. The following sections give a brief overview and scope of each functional block.

1.1.1 Capabilities

Capabilities test cases cover verification of getting Access Control Service capabilities. It means that the following commands are covered by these test cases:

- GetServices (Device Management Service)
- GetServiceCapabilities

1.1.2 Access Point

Access Point test cases cover verification of getting Access Point list, information, state and to enable/disable access point. It means that the following commands are covered by these test cases:

- GetAccessPointInfo
- GetAccessPointInfoList
- GetAccessPointState
- EnableAccessPoint
- DisableAccessPoint

1.1.3 Area

Area test cases cover verification of getting Area list and information. It means that the following commands are covered by these test cases:

- GetAreaInfo
- GetAreaInfoList

1.1.4 External Authorization

External Authorization test cases cover verification of external authorization procedure. It means that the following commands and events are covered by these test cases:

- ExternalAuthorization
- tns1:AccessControl/AccessGranted/Anonymous
- tns1:AccessControl/AccessGranted/Credential
- tns1:AccessControl/Denied/Anonymous
- tns1:AccessControl/Denied/Credential
- tns1:AccessControl/Request/Anonymous
- tns1:AccessControl/Request/Timeout
- tns1:AccessControl/Request/Credential

1.1.5 Property Events

Property events test cases cover verification of property events provided by Access Control Service. It means that the following commands and events are covered by these test cases:

- EnableAccessPoint
- DisableAccessPoint
- tns1:AccessPoint/State/Enabled

1.1.6 Access granted events

Access Granted events test cases cover verification of access granted events provided by Access Control Service. It means that the following events are covered by these test cases:

- tns1:AccessControl/AccessGranted/Anonymous
- tns1:AccessControl/AccessGranted/Credential

1.1.7 Access taken events

Access taken events test cases cover verification of access taken events provided by Access Control Service. It means that the following events are covered by these test cases:

- tns1:AccessControl/AccessTaken/Anonymous
- tns1:AccessControl/AccessTaken/Credential

1.1.8 Access not taken events

Access not taken events test cases cover verification of access not taken events provided by Access Control Service. It means that the following events are covered by these test cases:

- tns1:AccessControl/AccessNotTaken/Anonymous
- tns1:AccessControl/AccessNotTaken/Credential

1.1.9 Access denied events

Access denied events test cases cover verification of access denied events provided by Access Control Service. It means that the following events are covered by these test cases:

- tns1:AccessControl/Denied/Anonymous
- tns1:AccessControl/Denied/Credential
- tns1:AccessControl/Denied/CredentialNotFound/Card

1.1.10 Duress events

Duress events test cases cover verification of duress events provided by Access Control Service. It means that the following events are covered by the following test case:

- tns:AccessControl/Duress

1.1.11 Consistency

Consistency test cases cover verification of consistency between different entities and commands. It means that consistency between the following entities is covered by the following test case:

- Access Point Info and Get Area Info

1.1.12 Access Point Change Events

Access point change events test cases cover verification of access point change events provided by Access Control Service. It means that the following events are covered by these test cases:

- tns1:Configuration/AccessPoint/Change
- tns1:Configuration/AccessPoint/Removed

1.1.13 Area Change Events

Area change events test cases cover verification of area change events provided by Access Control Service. It means that the following events are covered by these test cases:

- tns1:Configuration/Area/Change
- tns1:Configuration/Area/Removed

2 Terms and Definitions

2.1 Definitions

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

Access Point A logical composition of a physical door and ID point(s) controlling access in one direction.

Access Point Disable If an Access Point is disabled, it will not be considered in the decision making process and no commands will be issued from that Access Point to the Door configured for that Access Point. When an Access Point is disabled, the associated ID Point may or may not be disabled or shut down. Clients may still be able to command the Door Controller to control associated door even though that door is also referenced by a disabled access point.

Credential A physical/tangible object, a piece of knowledge, or a facet of a person's physical being, that enables an individual access to a given physical facility or computer-based information system.

Credential Number A sequence of bytes uniquely identifying a credential at an access point.

Door A physical door, barrier, turnstile, etc which can be controlled remotely and restricts access between two areas. A door is usually equipped with an electronic lock and a door monitor.

Duress Forcing a person to provide access to a secure area against that person's wishes.

ID Point A device that converts reader signals to protocols recognized by an authorization engine. It can be card reader, REX, biometric reader etc.

2.2 Abbreviations

This section describes abbreviations used in this document.

HTTP	Hypertext Transfer (or Transport) Protocol
PACS	Physical Access Control System

3 Test Overview

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

3.1 Test Setup

3.1.1 Network Configuration for device under test

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

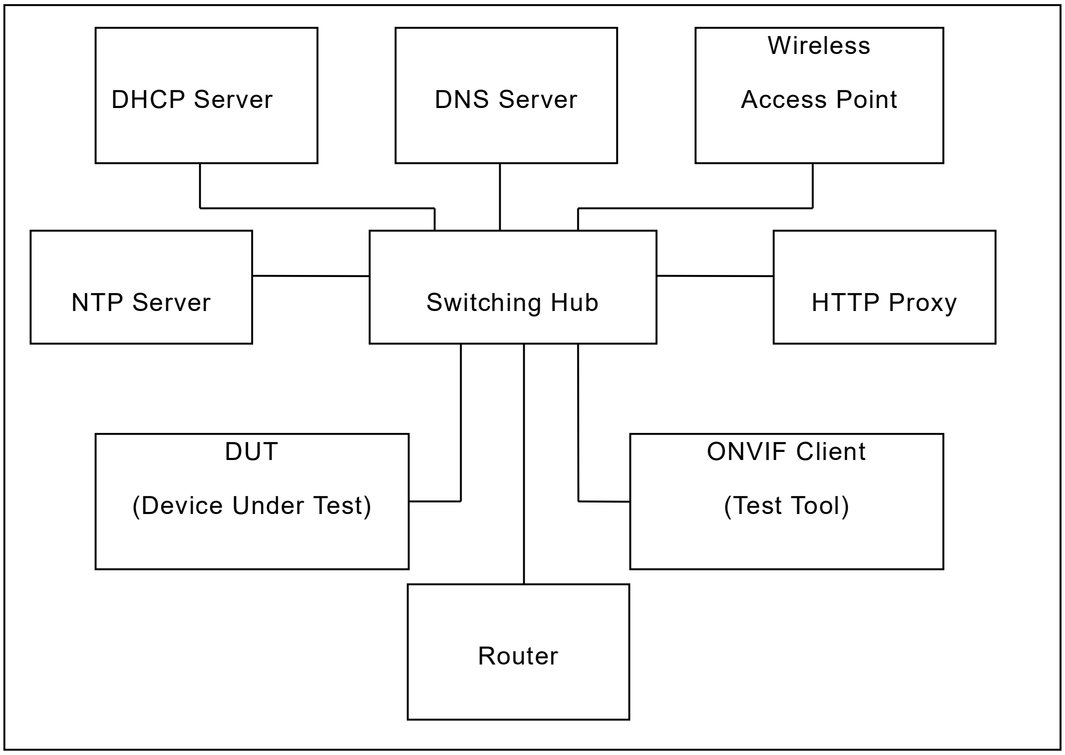


Figure 1: Test Configuration for DUT

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 connectivities among all the test equipment in the test environment. All devices should be connected to the Switching Hub.

Router: provides router advertisements for IPv6 configuration.

3.2 Prerequisites

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

- The DUT shall be configured with an IPv4 address.
- The DUT shall be IP reachable [in the test configuration].
- The DUT shall be able to be discovered by the ONVIF Device Test Tool.
- 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.
- The DUT time and ONVIF Device Test tool time shall be synchronized with each other either manually or by common NTP server.
- Test Operator shall configure Operation Delay properly so that it would have enough time to receive Notification messages for the following test cases for ONVIF Device Test Tool (see test description for more details):
    - 4.4.1 ACCESS CONTROL – ACCESS GRANTED TO ANONYMOUS (EXTERNAL AUTHORIZATION)
    - 4.4.2 ACCESS CONTROL – ACCESS DENIED TO ANONYMOUS (EXTERNAL AUTHORIZATION)
    - 4.4.3 ACCESS CONTROL – ACCESS TIMEOUT TO ANONYMOUS (EXTERNAL AUTHORIZATION)
    - 4.4.4 ACCESS CONTROL – ACCESS GRANTED WITH CREDENTIAL (EXTERNAL AUTHORIZATION)
    - 4.4.5 ACCESS CONTROL – ACCESS DENIED WITH CREDENTIAL (EXTERNAL AUTHORIZATION)
    - 4.4.6 ACCESS CONTROL – ACCESS TIMEOUT WITH CREDENTIAL (EXTERNAL AUTHORIZATION)
    - 4.5.1 ACCESS CONTROL – ACCESS POINT ENABLED EVENT
    - 4.5.2 ACCESS CONTROL – ACCESS POINT ENABLED EVENT STATE CHANGE
    - 4.6.1 ACCESS CONTROL – ACCESS GRANTED TO ANONYMOUS EVENT
    - 4.6.2 ACCESS CONTROL – ACCESS GRANTED WITH CREDENTIAL EVENT
    - 4.7.1 ACCESS CONTROL – ACCESS TAKEN BY ANONYMOUS EVENT
    - 4.7.2 ACCESS CONTROL – ACCESS TAKEN WITH CREDENTIAL EVENT
    - 4.8.1 ACCESS CONTROL – ACCESS NOT TAKEN BY ANONYMOUS EVENT
    - 4.8.2 ACCESS CONTROL – ACCESS NOT TAKEN WITH CREDENTIAL EVENT

  - 4.9.1 ACCESS CONTROL – ACCESS DENIED TO ANONYMOUS EVENT
  - 4.9.2 ACCESS CONTROL – ACCESS DENIED WITH CREDENTIAL EVENT
  - 4.9.3 ACCESS CONTROL – ACCESS DENIED WITH CREDENTIAL EVENT (CREDENTIAL NOT FOUND – CARD)
  - 4.10.1 ACCESS CONTROL – DURESS
  - 4.12.1 ACCESS CONTROL – ADD OR CHANGE ACCESS POINT EVENT
  - 4.12.2 ACCESS CONTROL – REMOVE ACCESS POINT EVENT
  - 4.13.1 ACCESS CONTROL – ADD OR CHANGE AREA EVENT
  - 4.13.2 ACCESS CONTROL – REMOVE AREA EVENT

- At least one Access Point is configured and added to the DUT.
- At least one Access Point with Enable/Disable capability is configured and added to the DUT, if Enable/Disable capability is supported by the DUT.
- At least one Access Point with Duress capability is configured and added to the DUT, if Duress capability is supported by the DUT.
- At least one Access Point with Access Taken capability is configured and added to the DUT, if Access Taken capability is supported by the DUT.
- At least one Access Point with External Authorization capability is configured and added to the DUT, if External Authorization capability is supported by the DUT.
- At least one Access Point with Anonymous Access capability is configured and added to the DUT, if Anonymous Access capability is supported by the DUT.
- At least one Access Point with Anonymous Access capability and External Authorization capability is configured and added to the DUT, if Anonymous Access capability and External Authorization capability are supported by the DUT.
- At least one Access Point with Anonymous Access capability and Access Taken capability is configured and added to the DUT, if Anonymous Access capability and Access Taken capability are supported by the DUT.
- At least one Access Point with tns1:AccessControl/Denied/CredentialNotFound/Card event capability is configured and added to the DUT, if tns1:AccessControl/Denied/CredentialNotFound/Card capability is supported by the DUT.
- At least one Area is configured and added to the DUT, if it is possible for the DUT.

3.3 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.

3.3.1 Capabilities

DUT shall give the Access Control Service entry point by GetServices command, if DUT supports this service. Otherwise, these test cases will be skipped.

DUT shall support GetServices and GetServiceCapabilities command.

Please, refer to Section 4.1 for Capabilities Test Cases.

3.3.2 Access Point

DUT shall give the Access Control Service entry point by GetServices command, if DUT supports this service. Otherwise, these test cases will be skipped.

DUT shall support GetServices and GetServiceCapabilities command.

DUT shall support at least one Access Point. And also at least one Access Point shall be added and configured on device.

DUT shall support the following commands:

- GetAccessPointInfo
- GetAccessPointInfoList
- GetAccessPointState

DUT shall return capabilities for each specific access point. If DUT returns Disable Access Point capability as supported by access point, then DUT shall support the following commands for this access point:

- DisableAccessPoint
- EnableAccessPoint

DUT shall not return any fault, if GetAccessPointInfo was invoked for nonexistent Access Point token. Such tokens shall be ignored.

DUT shall not return more items in GetAccessPointInfo and GetAccessPointInfoList responses than specified in service capabilities by MaxLimit.

DUT shall not return more items in GetAccessPointInfoList response than specified by Limit parameter in request.

DUT shall return SOAP 1.2 fault message (InvalidArgs/TooManyItems), if more items than MaxLimit were requested by GetAccessPointInfo command.

DUT should return SOAP 1.2 fault message (InvalidArgVal/NotFound), if GetAccessPointState, EnableAccessPoint or DisableAccessPoint command was invoked for nonexistent Access Point token.

DUT should return SOAP 1.2 fault message (ActionNotSupported/NotSupported), if EnableAccessPoint or DisableAccessPoint command was invoked for Access Point without Disable Access Point capability token.

Please, refer to Section 4.2 for Access Point Test Cases.

3.3.3 Area

DUT shall give the Access Control Service entry point by GetServices command, if DUT supports this service. Otherwise, these test cases will be skipped.

DUT shall support GetServiceCapabilities command.

If DUT supports at least one Area, one Area shall be added and configured on device.

DUT shall support the following commands:

- GetAreaInfo
- GetAreaInfoList

DUT shall not return any fault, if GetAreaInfo was invoked for nonexistent Area token. Such tokens shall be ignored.

DUT shall not return more items in GetAreaInfo and GetAreaInfoList responses than specified in service capabilities by MaxLimit.

DUT shall not return more items in GetAreaInfoList response than specified by Limit parameter in request.

DUT shall return SOAP 1.2 fault message (InvalidArgs/TooManyItems), if more items than MaxLimit were requested by GetAreaInfo command.

Please, refer to Section 4.3 for Area Test Cases.

3.3.4 External Authorization

DUT shall give the Access Control Service entry point and Event Service entry point by GetServices command, if DUT supports Access Control service. Otherwise, these test cases will be skipped.

DUT shall support GetServiceCapabilities command.

DUT shall return capabilities for each specific access point.

If DUT returns External Authorization capability as supported by access point, then DUT shall support the following commands and events for this access point:

- ExternalAuthorization
- tns1:AccessControl/AccessGranted/Credential
- tns1:AccessControl/Denied/Credential
- tns1:AccessControl/Request/Timeout
- tns1:AccessControl/Request/Credential

If DUT returns External Authorization capability and Anonymous Access capability as supported by access point, then DUT shall support the following events for this access point:

- tns1:AccessControl/AccessGranted/Anonymous

- tns1:AccessControl/Denied/Anonymous
- tns1:AccessControl/Request/Anonymous

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

DUT shall support Pull Point Subscription and Topic Expression filter.

DUT should return SOAP 1.2 fault message (InvalidArgVal/NotFound), if ExternalAuthorization command was invoked for nonexistent Access Point token.

DUT should return SOAP 1.2 fault message (ActionNotSupported/NotSupported), if ExternalAuthorization command was invoked for Access Point without External Authorization capability token.

A test operator shall manually invoke the following events if required:

- tns1:AccessControl/Request/Timeout
- tns1:AccessControl/Request/Credential
- tns1:AccessControl/Request/Anonymous

Please, refer to Section 4.4 for External Authorization Test Cases.

3.3.5 Property Events

DUT shall give the Access Control Service entry point and Event Service entry point by GetServices command, if DUT supports Access Control service. Otherwise these test cases will be skipped.

DUT shall support GetServiceCapabilities command.

DUT shall support following property events:

- tns1:AccessPoint/State/Enabled

DUT shall return capabilities for each specific access point. If DUT returns Disable Access Point capability as supported by access point, then DUT shall support the following commands for this access point:

- DisableAccessPoint
- EnableAccessPoint

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.

DUT shall generate property events with current state after corresponding properties were changed.

Please, refer to Section 4.5for Property Events Test Cases.

3.3.6 Access granted events

DUT shall give the Access Control Service entry point and Event Service entry point by GetServices command, if DUT supports Access Control service. Otherwise these test cases will be skipped.

DUT shall support GetServiceCapabilities command.

DUT shall return capabilities for each specific access point.

DUT shall support the following events for each access point:

- tns1:AccessControl/AccessGranted/Credential

If DUT returns Anonymous Access capability as supported by access point, then DUT shall support the following events for this access point:

- tns1:AccessControl/AccessGranted/Anonymous

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

DUT shall support Pull Point Subscription and Topic Expression filter.

A test operator shall manually invoke the following events if required:

- tns1:AccessControl/AccessGranted/Anonymous
- tns1:AccessControl/AccessGranted/Credential

Please, refer to Section 4.6 for Access granted events Test Cases.

3.3.7 Access taken events

DUT shall give the Access Control Service entry point and Event Service entry point by GetServices command, if DUT supports Access Control service. Otherwise, these test cases will be skipped.

DUT shall support GetServiceCapabilities command.

DUT shall return capabilities for each specific access point.

DUT shall support the following events for each access point:

- tns1:AccessControl/AccessGranted/Credential

If DUT returns Anonymous Access capability as supported by access point, then DUT shall support the following events for this access point:

- tns1:AccessControl/AccessGranted/Anonymous

If DUT returns Access Taken capability as supported by access point, then DUT shall support the following events for this access point:

- tns1:AccessControl/AccessTaken/Credential

If DUT returns Anonymous Access capability and Access Taken capability as supported by access point, then DUT shall support the following events for this access point:

- tns1:AccessControl/AccessTaken/Anonymous

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

DUT shall support Pull Point Subscription and Topic Expression filter.

A test operator shall manually invoke the following events if required:

- tns1:AccessControl/AccessGranted/Anonymous
- tns1:AccessControl/AccessGranted/Credential
- tns1:AccessControl/AccessTaken/Anonymous
- tns1:AccessControl/AccessTaken/Credential

Please, refer to Section 4.7 for Access taken events Test Cases.

3.3.8 Access not taken events

DUT shall give the Access Control Service entry point and Event Service entry point by GetServices command, if DUT supports Access Control service. Otherwise, these test cases will be skipped.

DUT shall support GetServiceCapabilities command.

DUT shall return capabilities for each specific access point.

DUT shall support the following events for each access point:

- tns1:AccessControl/AccessGranted/Credential

If DUT returns Anonymous Access capability as supported by access point, then DUT shall support the following events for this access point:

- tns1:AccessControl/AccessGranted/Anonymous

If DUT returns Access Taken capability as supported by access point, then DUT shall support the following events for this access point:

- tns1:AccessControl/AccessNotTaken/Credential

If DUT returns Anonymous Access capability and Access Taken capability as supported by access point, then DUT shall support the following events for this access point:

- tns1:AccessControl/AccessNotTaken/Anonymous

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

DUT shall support Pull Point Subscription and Topic Expression filter.

A test operator shall manually invoke the following events if required:

- tns1:AccessControl/AccessGranted/Anonymous
- tns1:AccessControl/AccessGranted/Credential
- tns1:AccessControl/AccessNotTaken/Anonymous
- tns1:AccessControl/AccessNotTaken/Credential

Please, refer to Section 4.8 for Access not taken events Test Cases.

3.3.9 Access denied events

DUT shall give the Access Control Service entry point and Event Service entry point by GetServices command, if DUT supports Access Control service. Otherwise, these test cases will be skipped.

DUT shall support GetServiceCapabilities command.

DUT shall return capabilities for each specific access point.

DUT shall support the following events for each access point:

- tns1:AccessControl/Denied/Credential

If DUT returns tns1:AccessControl/Denied/CredentialNotFound/Card as supported, DUT shall support the following event for at least one configured access point:

- tns1:AccessControl/Denied/CredentialNotFound/Card

If DUT returns Anonymous Access capability as supported by access point, then DUT shall support the following events for this access point:

- tns1:AccessControl/Denied/Anonymous

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

DUT shall support Pull Point Subscription and Topic Expression filter.

A test operator shall manually invoke the following events if required:

- tns1:AccessControl/Denied/Anonymous
- tns1:AccessControl/Denied/Credential
- tns1:AccessControl/Denied/CredentialNotFound/Card

Please, refer to Section 4.9 for Access denied events Test Cases.

3.3.10 Duress events

DUT shall give the Access Control Service entry point and Event Service entry point by GetServices command, if DUT supports Access Control service. Otherwise, these test cases will be skipped.

DUT shall support GetServiceCapabilities command.

DUT shall return capabilities for each specific access point.

If DUT returns Duress capability as supported by access point, then DUT shall support the following events for this access point:

- tns:AccessControl/Duress

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

DUT shall support Pull Point Subscription and Topic Expression filter.

A test operator shall manually invoke the following events if required:

- tns:AccessControl/Duress

Please, refer to Section 4.10 for Duress events Test Cases.

3.3.11 Consistency

DUT shall give the Access Control Service entry point by GetServices command, if DUT supports this service. Otherwise, these test cases will be skipped.

DUT shall support at least one Access Point. And also at least one Access Point shall be added and configured on device.

If DUT supports at least one Area, at least one Area shall be added and configured on device.

DUT shall support the following commands:

- GetAccessPointInfoList
- GetAreaInfoList

DUT shall not return other Area tokens, than listed in GetAreaInfoList responses, in AreaTo and AreaFrom fields of GetAccessPointInfo's.

Please, refer to Section 4.11 for Consistency Test Cases.

3.3.12 Access Point Change Events

DUT shall give the Access Control Service entry point and Event Service entry point by GetServices command, if DUT supports Access Control service. Otherwise, these test cases will be skipped.

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

DUT shall support Pull Point Subscription and Topic Expression filter.

If DUT supports adding or changing access points, then DUT shall support the following events:

- tns1:Configuration/AccessPoint/Change

If DUT returns tns1:Configuration/AccessPoint/Change topic in GetEventProperties command, then DUT should have possibility to invoke it.

If DUT supports removing access points, then DUT shall support the following events:

- tns1:Configuration/AccessPoint/Removed

If DUT returns tns1:Configuration/AccessPoint/Removed topic in GetEventProperties command, then DUT should have possibility to invoke it.

A test operator shall manually invoke the following events if required:

- tns1:Configuration/AccessPoint/Change
- tns1:Configuration/AccessPoint/Removed

Please, refer to Section 4.12 for Access Point Change events Test Cases.

3.3.13 Area Change Events

DUT shall give the Access Control Service entry point and Event Service entry point by GetServices command, if DUT supports Access Control service. Otherwise, these test cases will be skipped.

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

DUT shall support Pull Point Subscription and Topic Expression filter.

If DUT supports adding or changing areas, then DUT shall support the following events:

- tns1:Configuration/Area/Change

If DUT returns tns1:Configuration/Area/Change topic in GetEventProperties command, then DUT should have possibility to invoke it.

If DUT supports removing areas, then DUT shall support the following events:

- tns1:Configuration/Area/Removed

If DUT returns tns1:Configuration/Area/Removed topic in GetEventProperties command, then DUT should have possibility to invoke it.

A test operator shall manually invoke the following events if required:

- tns1:Configuration/Area/Change
- tns1:Configuration/Area/Removed

Please, refer to Section 4.13 for Area Change events Test Cases.

4 Access Control Test Cases

4.1 Capabilities

4.1.1 ACCESS CONTROL SERVICE CAPABILITIES

Test Label: Access Control Capabilities Verification.

Test Case ID: ACCESSCONTROL-1-1-1

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

Command under test: GetServiceCapabilities (for Access Control Service)

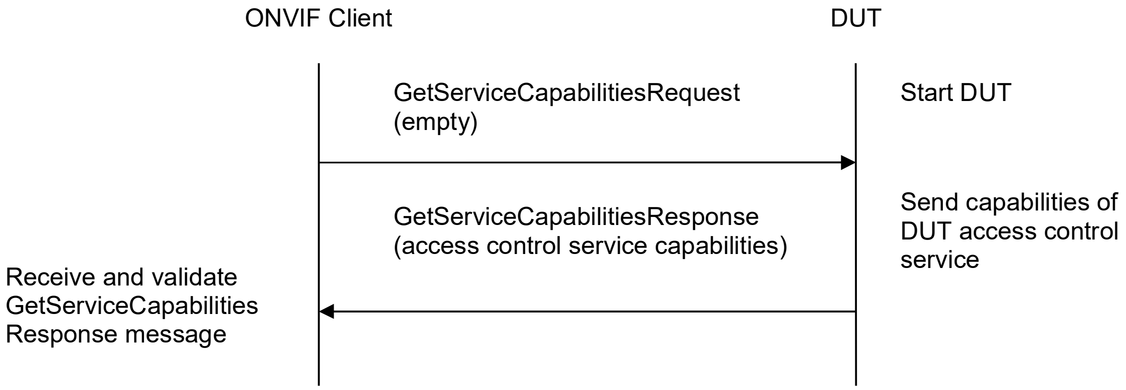
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify DUT Access Control Service Capabilities.

Pre-Requisite: Access Control Service was received from the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
GetServiceCapabilitiesRequest (empty)
Start DUT
GetServiceCapabilitiesResponse (access control service capabilities)
Send capabilities of DUT access control service
Receive and validate GetServiceCapabilities Response message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve DUT access control service capabilities.

4. Verify the GetServiceCapabilitiesResponse message from the DUT.

Test Result:

PASS –

The DUT passes all assertions.

FAIL –

The DUT did not send valid GetServiceCapabilitiesResponse.

4.1.2 GET SERVICES AND GET ACCESS CONTROL SERVICE CAPABILITIES CONSISTENCY

Test Label: Get Services and Access Control Service Capabilities Consistency Verification.

Test Case ID: ACCESSCONTROL-1-1-2

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

Command under test: GetServices, GetServiceCapabilities (for Access Control Service)

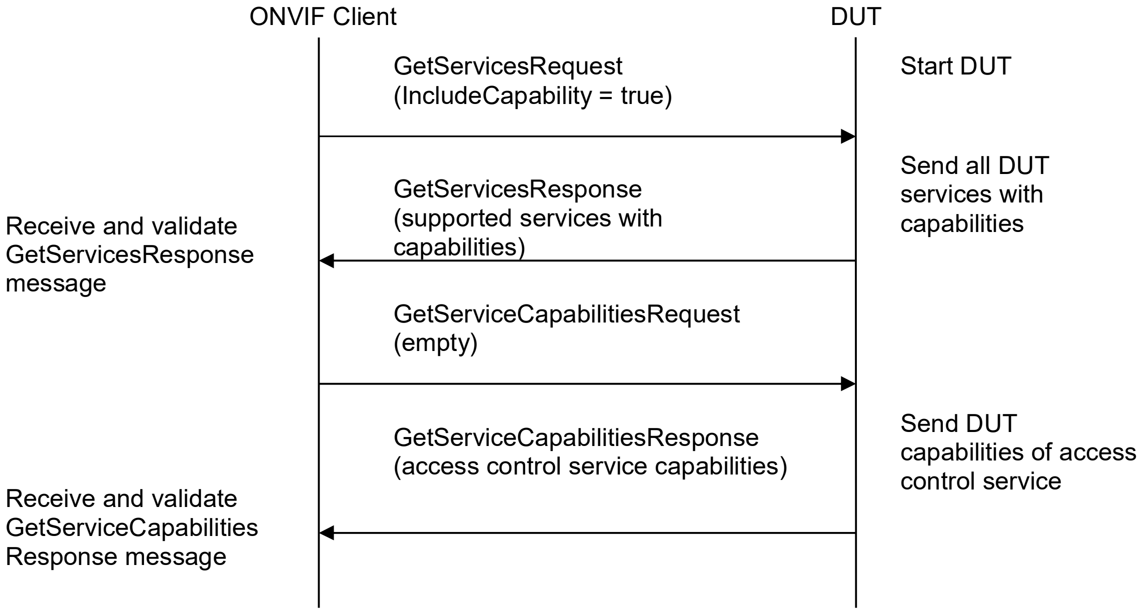
WSDL Reference: devicemgmt.wsdl, accesscontrol.wsdl

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

Pre-Requisite: None.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
GetServicesRequest (IncludeCapability = true)
Start DUT
GetServicesResponse (supported services with capabilities)
Send all DUT services with capabilities
Receive and validate GetServicesResponse message
GetServiceCapabilitiesRequest (empty)
GetServiceCapabilitiesResponse (access control service capabilities)
Send DUT capabilities of access control service
Receive and validate GetServiceCapabilities Response message

Test Procedure:

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

5. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve DUT Access Control service capabilities.

6. Verify the GetServiceCapabilitiesResponse message from the DUT.

Test Result:

PASS –

The DUT passes all assertions.

FAIL –

The DUT did not send valid GetServicesResponse message.

The DUT did not send valid GetServiceCapabilitiesResponse message.

The DUT sent different Capabilities in GetServicesResponse message and in GetServiceCapabilitiesResponse message.

Note: Service will be defined as Access Controller service if it has Namespace element that is equal to “http://www.onvif.org/ver10/accesscontrol/wsdl”.

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

- Values of MaxLimit attribute are different.

4.2 Access Point

4.2.1 GET ACCESS POINT INFO

Test Label: Get Access Point Info Verification.

Test Case ID: ACCESSCONTROL-2-1-1

ONVIF Core Specification Coverage: GetAccessPointInfo (ONVIF Access Control Service Specification)

Command under test: GetAccessPointInfo

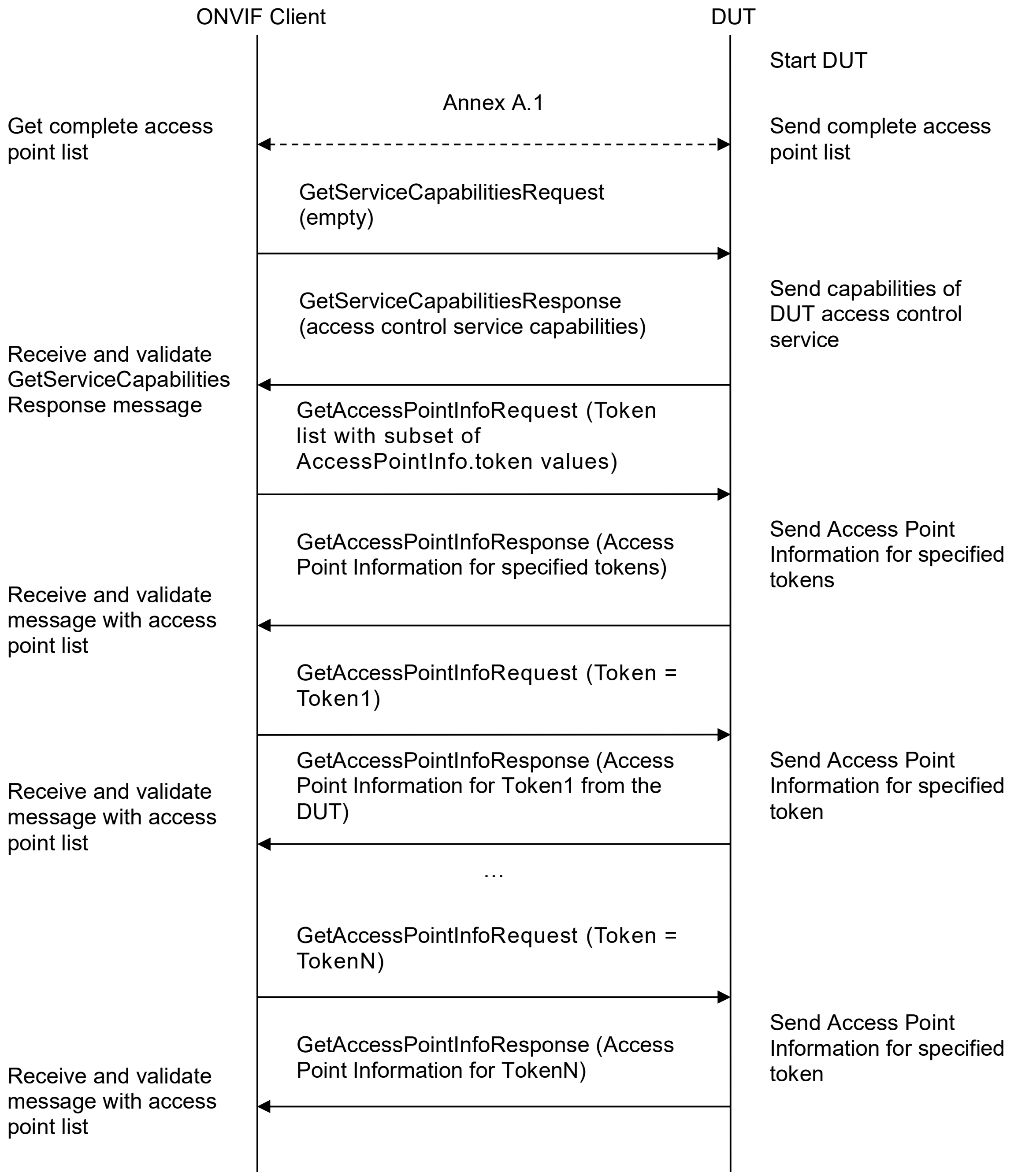
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Get Access Point Info.

Pre-Requisite: Access Control Service was received from the DUT. At least one Access Point is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetServiceCapabilitiesRequest (empty)
GetServiceCapabilitiesResponse (access control service capabilities)
Send capabilities of DUT access control service
Receive and validate GetServiceCapabilities Response message
GetAccessPointInfoRequest (Token list with subset of AccessPointInfo.token values)
GetAccessPointInfoResponse (Access Point Information for specified tokens)
Send Access Point Information for specified tokens
Receive and validate message with access point list
GetAccessPointInfoRequest (Token = Token1)
GetAccessPointInfoResponse (Access Point Information for Token1 from the DUT)
Send Access Point Information for specified token
Receive and validate message with access point list
…
GetAccessPointInfoRequest (Token = TokenN)
GetAccessPointInfoResponse (Access Point Information for TokenN)
Send Access Point Information for specified token
Receive and validate message with access point list

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. Get complete list of access points from the DUT (see Annex A.1).

4. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve MaxLimit capability.

5. Verify the GetServiceCapabilitiesResponse message from the DUT.

6. ONVIF Client will invoke GetAccessPointInfoRequest message (**Token** list with subset of AccessPointInfo.token values from a complete list of access points at step 3 with token number equal to MaxLimit) to retrieve a subset of Access Point Information from the DUT.

7. Verify the GetAccessPointInfoResponse message from the DUT.

8. Verify that all requested **AccessPointInfo** items are in GetAccessPointInfoResponse message.

9. ONVIF Client will invoke GetAccessPointRequest message (**Token = Token1**, where Token1 is the first token from the complete list of access points at step 3) to retrieve AccessPoint Information for specified token from the DUT.

10. Verify the GetAccessPointInfoResponse message from the DUT.

11. Verify that GetAccessPointInfoResponse message contains **AccessPointInfo** only for specified token.

12. Repeat steps 9-11 for all other tokens from a complete list of access points at step 3.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetAccessPointInfoResponse message.

The DUT returned GetAccessPointInfoResponse message that contains at least two AccessPointInfo items with equal tokens.

The DUT did not return requested AccessPointInfo items in GetAccessPointInfoResponse message for step 7 or 10.

The DUT returned more items than requested in GetAccessPointInfoResponse message for step 7 or 10.

The DUT did not return at least one Access Point at step 3.

The DUT did not send valid GetServiceCapabilitiesResponse.

Note: If the DUT did not return any AccessPointInfo on complete list of access points at step 3, skip steps 4-12, fail test and go to the next test.

Note: In case MaxLimit is equal to 0, 1 will be used instead in requests.

4.2.2 GET ACCESS POINT INFO WITH INVALID TOKEN

Test Label: Get Access Point Info Verification with Invalid Token.

Test Case ID: ACCESSCONTROL-2-1-2

ONVIF Core Specification Coverage: GetAccessPointInfo (ONVIF Access Control Service Specification)

Command under test: GetAccessPointInfo

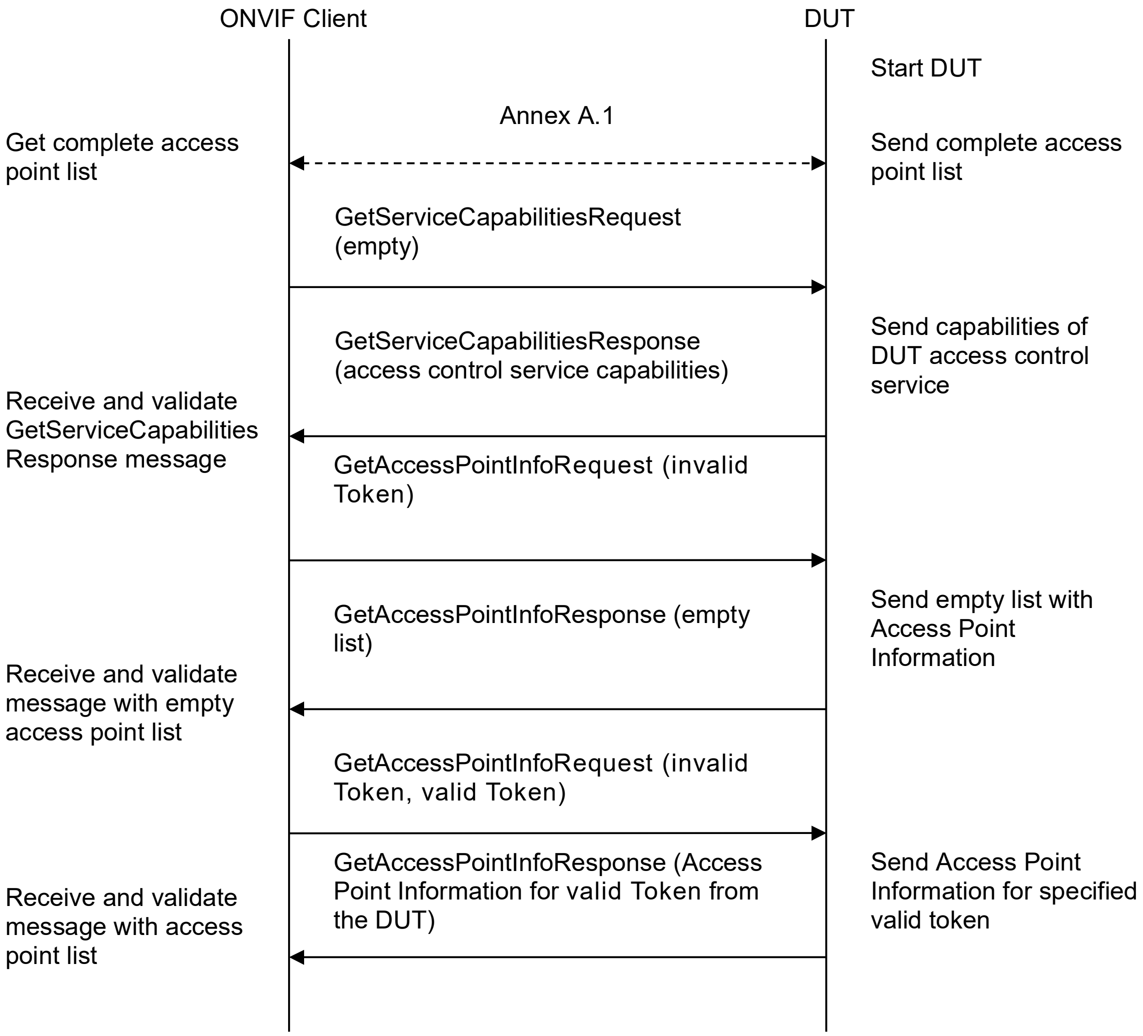
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Get Access Point Info with invalid Token.

Pre-Requisite: Access Control Service was received from the DUT. At least one Access Point is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetServiceCapabilitiesRequest (empty)
GetServiceCapabilitiesResponse (access control service capabilities)
Send capabilities of DUT access control service
Receive and validate GetServiceCapabilities Response message
GetAccessPointInfoRequest (invalid Token)
GetAccessPointInfoResponse (empty list)
Send empty list with Access Point Information
Receive and validate message with empty access point list
GetAccessPointInfoRequest (invalid Token, valid Token)
GetAccessPointInfoResponse (Access Point Information for valid Token from the DUT)
Send Access Point Information for specified valid token
Receive and validate message with access point list

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get complete list of access points from the DUT (see Annex A.1).
4. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve MaxLimit capability.
5. Verify the GetServiceCapabilitiesResponse message from the DUT.
6. ONVIF Client will invoke GetAccessPointInfoRequest message **(invalid Token**).
7. Verify the GetAccessPointInfoResponse message from the DUT. Check that empty list was returned.
8. ONVIF Client will invoke GetAccessPointInfoRequest message **(invalid Token, valid Token**).
9. Verify the GetAccessPointInfoResponse message from the DUT. Check that list which contains AccessPoint for valid Token only was returned.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetAccessPointInfoResponse message.

The DUT returned GetAccessPointInfoResponse message that contains at least two AccessPointInfo items with equal tokens.

The DUT did not send valid GetAccessPointInfoResponse message.

The DUT returned GetAccessPointInfoResponse message that contains any AccessPointInfo items for step 7.

The DUT returned GetAccessPointInfoResponse message that contains any AccessPointInfo items other than item for valid Token or does not contains AccessPointInfo item for valid token step 8.

The DUT did not return at least one Access Point at step 3.

The DUT did not send valid GetServiceCapabilitiesResponse.

Note: If MaxLimit is less than 2 skip steps 8-9.

4.2.3 GET ACCESS POINT INFO LIST – LIMIT

Test Label: Get Access Point Info List Verification with Limit.

Test Case ID: ACCESSCONTROL-2-1-3

ONVIF Core Specification Coverage: GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: GetAccessPointInfoList

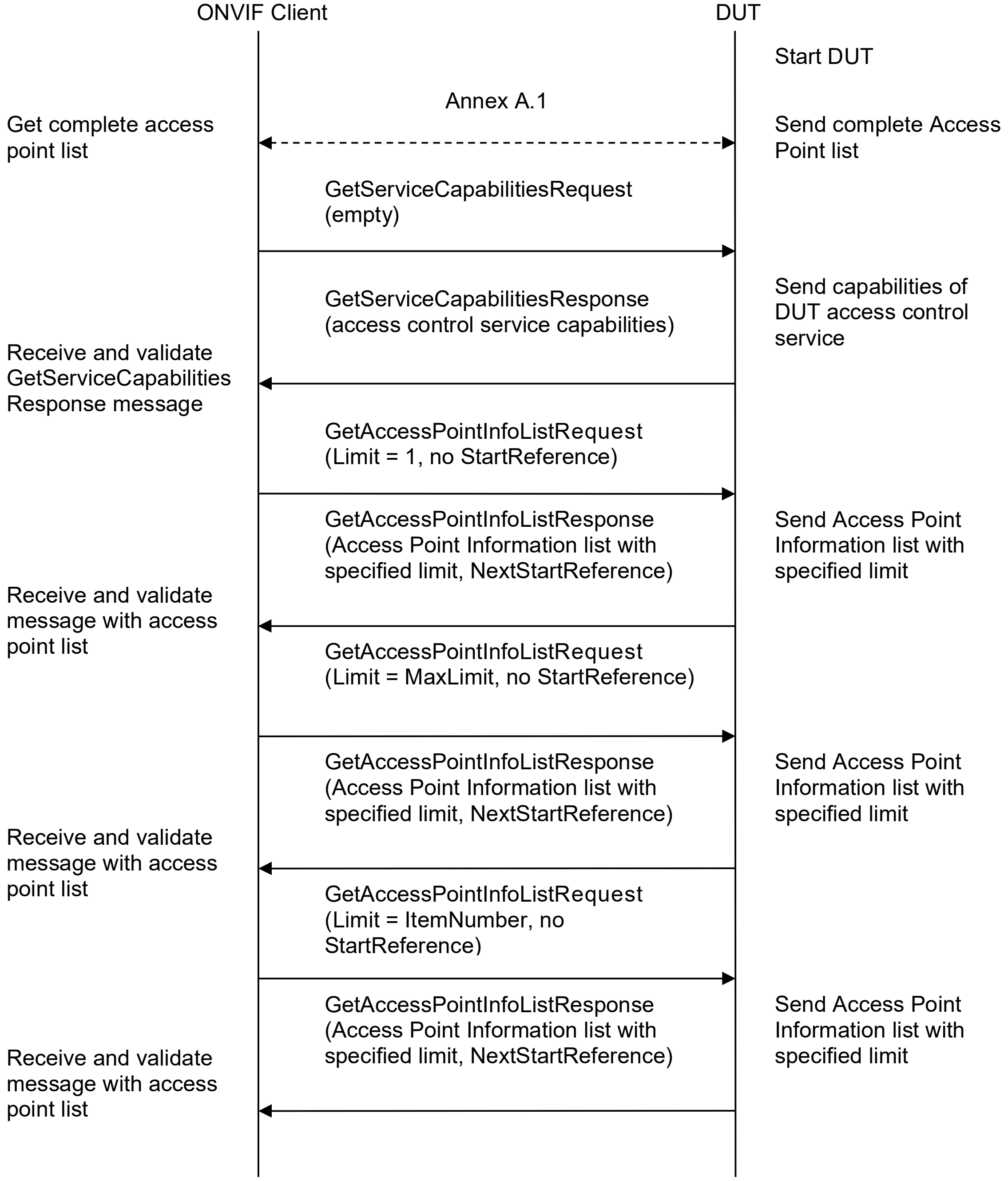
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Get Access Point Info List using Limit.

Pre-Requisite: Access Control Service was received from the DUT. At least one Access Point is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete Access Point list
GetServiceCapabilitiesRequest (empty)
GetServiceCapabilitiesResponse (access control service capabilities)
Send capabilities of DUT access control service
Receive and validate GetServiceCapabilities Response message
GetAccessPointInfoListRequest (Limit = 1, no StartReference)
GetAccessPointInfoListResponse (Access Point Information list with specified limit, NextStartReference)
Send Access Point Information list with specified limit
Receive and validate message with access point list
GetAccessPointInfoListRequest (Limit = MaxLimit, no StartReference)
GetAccessPointInfoListResponse (Access Point Information list with specified limit, NextStartReference)
Send Access Point Information list with specified limit
Receive and validate message with access point list
GetAccessPointInfoListRequest (Limit = ItemNumber, no StartReference)
GetAccessPointInfoListResponse (Access Point Information list with specified limit, NextStartReference)
Send Access Point Information list with specified limit
Receive and validate message with access point list

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get complete list of access points from the DUT (see Annex A.1)

4. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve MaxLimit capability.

5. Verify the GetServiceCapabilitiesResponse message from the DUT.

6. ONVIF Client will invoke GetAccessPointInfoListRequest message (**Limit = 1, no StartReference**) to retrieve the first item on the list of Access Point Information from the DUT.

7. Verify the GetAccessPointInfoListResponse message from the DUT.

8. Verify that GetAccessPointInfoListResponse message at step 7 does not contain number of items greater than 1. Verify that GetAccessPointInfoListResponse message at step 5 does not contain number of items greater than MaxLimit.

9. ONVIF Client will invoke GetAccessPointInfoListRequest message (**Limit = MaxLimit, no StartReference**) to retrieve the first part of Access Point Information list from the DUT and is limited by MaxLimit.

10. Verify the GetAccessPointInfoListResponse message from the DUT.

11. Verify that GetAccessPointInfoListResponse message at step 10 does not contain number of items greater than MaxLimit.

12. ONVIF Client will invoke GetAccessPointInfoListRequest message (**Limit = ItemNumber, no StartReference,** where ItemNumber is between 1 and minimal value of MaxLimit and total number of access points) to retrieve sublist of Access Point Information limited by ItemNumber from the DUT.

13. Verify the GetAccessPointInfoListResponse message from the DUT.

14. Verify that GetAccessPointInfoListResponse message at step 13 does not contain number of items greater than ItemNumber. Verify that GetAccessPointInfoListResponse message at step 13 does not contain number of items greater than MaxLimit.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetAccessPointInfoListResponse message.

The DUT did not return at least one Access Point on step 3.

The DUT returned more AccessPointInfo items in GetAccessPointInfoListResponse message than it was specified in Limit parameter.

The DUT returned more AccessPointInfo items in GetAccessPointInfoListResponse message than MaxLimit.

The DUT did not send valid GetServiceCapabilitiesResponse.

Note: If the DUT did not return any AccessPointInfo on complete list of access points at step 3, skip steps 4-14, fail test and go to the next test.

Note: In case MaxLimit is equal to 0, 1 will be used instead for request.

4.2.4 GET ACCESS POINT INFO LIST – START REFERENCE AND LIMIT

Test Label: Get Access Point Info List Verification with StartReference and Limit.

Test Case ID: ACCESSCONTROL-2-1-4

ONVIF Core Specification Coverage: GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: GetAccessPointInfoList

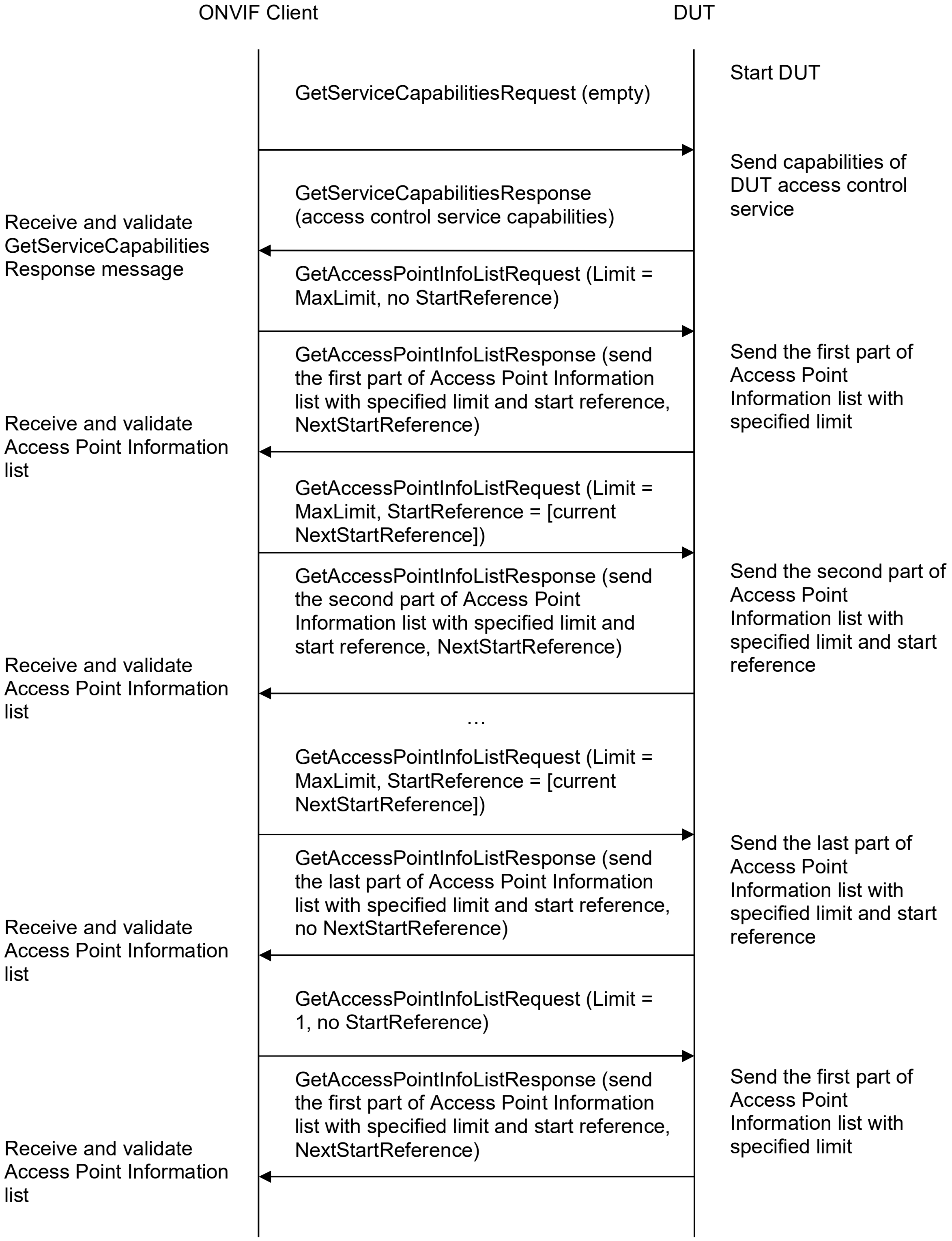
WSDL Reference: accesscontrol.wsdl

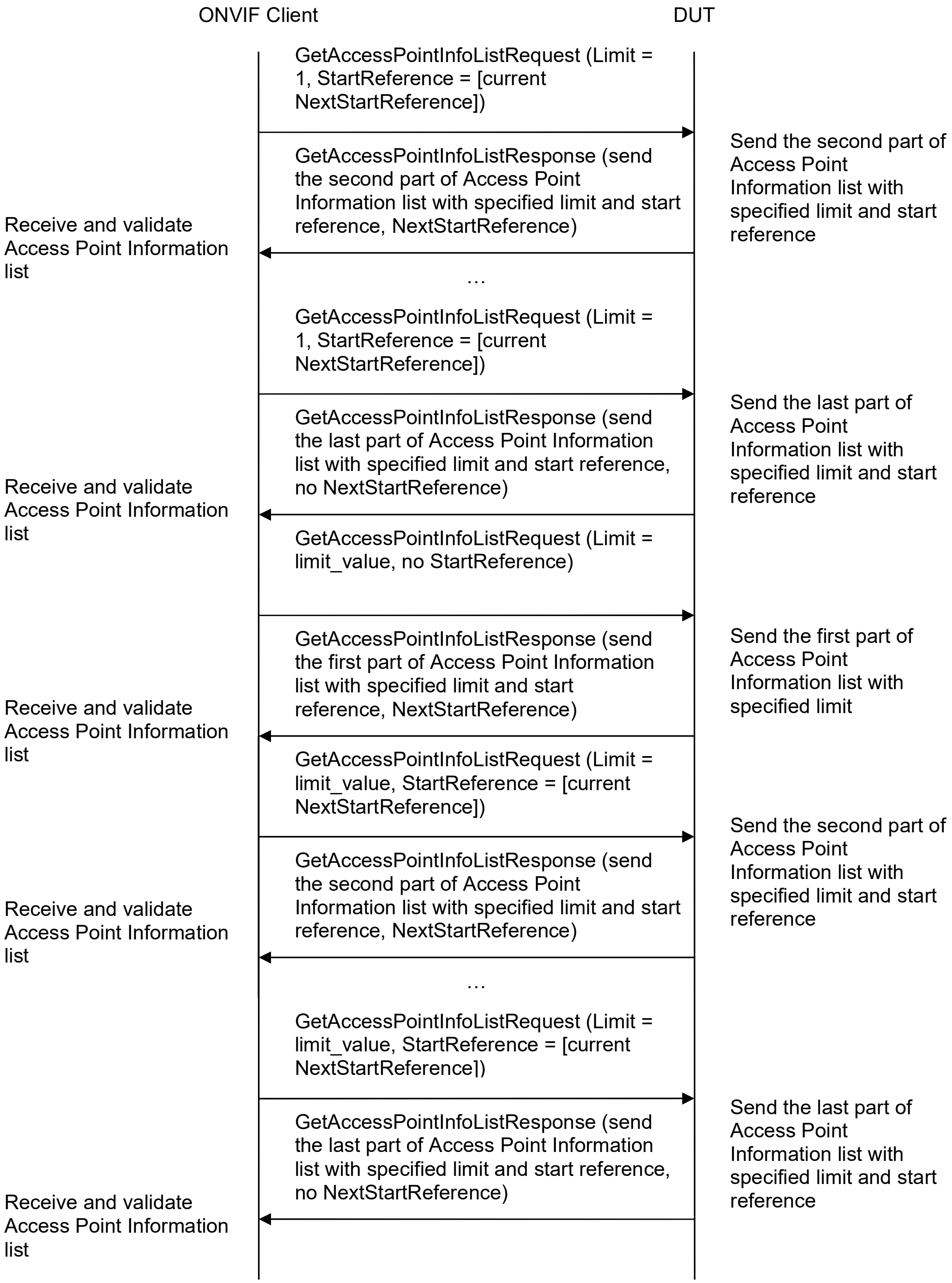
Test Purpose: To verify Get Access Point Info List using StartReference and Limit.

Pre-Requisite: Access Control Service was received from the DUT. At least one Access Point is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
GetServiceCapabilitiesRequest (empty)
Send capabilities of DUT access control service
GetServiceCapabilitiesResponse (access control service capabilities)
Receive and validate GetServiceCapabilities Response message
GetAccessPointInfoListRequest (Limit = MaxLimit, no StartReference)
GetAccessPointInfoListResponse (send the first part of Access Point Information list with specified limit and start reference, NextStartReference)
Send the first part of Access Point Information list with specified limit
Receive and validate Access Point Information list
GetAccessPointInfoListRequest (Limit = MaxLimit, StartReference = [current NextStartReference])
GetAccessPointInfoListResponse (send the second part of Access Point Information list with specified limit and start reference, NextStartReference)
Send the second part of Access Point Information list with specified limit and start reference
Receive and validate Access Point Information list
…
GetAccessPointInfoListRequest (Limit = MaxLimit, StartReference = [current NextStartReference])
GetAccessPointInfoListResponse (send the last part of Access Point Information list with specified limit and start reference, no NextStartReference)
Send the last part of Access Point Information list with specified limit and start reference
Receive and validate Access Point Information list
GetAccessPointInfoListRequest (Limit = 1, no StartReference)
GetAccessPointInfoListResponse (send the first part of Access Point Information list with specified limit and start reference, NextStartReference)
Send the first part of Access Point Information list with specified limit
Receive and validate Access Point Information list


ONVIF Client
DUT
GetAccessPointInfoListRequest (Limit = 1, StartReference = [current NextStartReference])
Send the second part of Access Point Information list with specified limit and start reference
GetAccessPointInfoListResponse (send the second part of Access Point Information list with specified limit and start reference, NextStartReference)
Receive and validate Access Point Information list
…
GetAccessPointInfoListRequest (Limit = 1, StartReference = [current NextStartReference])
Send the last part of Access Point Information list with specified limit and start reference
GetAccessPointInfoListResponse (send the last part of Access Point Information list with specified limit and start reference, no NextStartReference)
Receive and validate Access Point Information list
GetAccessPointInfoListRequest (Limit = limit_value, no StartReference)
Send the first part of Access Point Information list with specified limit
GetAccessPointInfoListResponse (send the first part of Access Point Information list with specified limit and start reference, NextStartReference)
Receive and validate Access Point Information list
GetAccessPointInfoListRequest (Limit = limit_value, StartReference = [current NextStartReference])
Send the second part of Access Point Information list with specified limit and start reference
GetAccessPointInfoListResponse (send the second part of Access Point Information list with specified limit and start reference, NextStartReference)
Receive and validate Access Point Information list
…
GetAccessPointInfoListRequest (Limit = limit_value, StartReference = [current NextStartReference])
Send the last part of Access Point Information list with specified limit and start reference
GetAccessPointInfoListResponse (send the last part of Access Point Information list with specified limit and start reference, no NextStartReference)
Receive and validate Access Point Information list

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve DUT access control service capabilities.

4. Verify the GetServiceCapabilitiesResponse (**MaxLimit**) message from the DUT.

5. ONVIF Client will invoke GetAccessPointInfoListRequest message (**Limit =** MaxLimit, no StartReference) to retrieve the first part of Access Point Information list from the DUT.

6. Verify the GetAccessPointInfoListResponse message from the DUT. Verify that GetAccessPointInfoListResponse message does not contain number of items greater than Limit. Verify that GetAccessPointInfoListResponse message does not contain number of items greater than MaxLimit.

7. If GetAccessPointInfoListResponse message contains NextStartReference, repeat steps 5-6 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting the complete access point list.

8. Verify that GetAccessPointInfoListResponse messages at step 6 does not contain AccessPointInfo items with the same tokens values all over the list, i.e. there shall be no GetAccessPointInfoListResponse messages containing the same value for AccessPointInfo items token.

9. ONVIF Client will invoke GetAccessPointInfoListRequest message (**Limit = 1**, no StartReference) to retrieve the first part of Access Point Information list from the DUT.

10. Verify the GetAccessPointInfoListResponse message from the DUT. Verify that GetAccessPointInfoListResponse message does not contain number of items greater than Limit. Verify that GetAccessPointInfoListResponse message does not contain number of items greater than MaxLimit.

11. If GetAccessPointInfoListResponse message contains NextStartReference, repeat steps 9-12 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting the complete access point list.

12. Verify that GetAccessPointInfoListResponse messages at step 10 does not contain AccessPointInfo items with the same tokens values all over the list, i.e. there shall be no GetAccessPointInfoListResponse messages containing the same value for AccessPointInfo items token.

13. Verify that the total list received during steps 5-7 has the same items as the list received during steps 9-11.

14. ONVIF Client will invoke GetAccessPointInfoListRequest message (**Limit = [limit_value]**, no StartReference, where limit_value is between 1 and MaxLimit) to retrieve the first part of Access Point Information list from the DUT.

15. Verify the GetAccessPointInfoListResponse message from the DUT. Verify that GetAccessPointInfoListResponse message does not contain number of items greater than Limit. Verify that GetAccessPointInfoListResponse message does not contain number of items greater than MaxLimit.

16. If GetAccessPointInfoListResponse message contains NextStartReference, repeat steps 14-17 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting the complete access point list.

17. Verify that GetAccessPointInfoListResponse messages at step 15 does not contain AccessPointInfo items with the same tokens values all over the list, i.e. there shall be no GetAccessPointInfoListResponse messages containing the same value for AccessPointInfo items token.

18. Verify that the total list received during steps 5-7 has the same items as the list received during steps 14-16.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetAccessPointInfoListResponse message.

The DUT returned GetAccessPointInfoListResponse message that contains wrong item list than requested according to StartReference and Limit values.

The DUT returned GetAccessPointInfoListResponse messages for step 6 contains at least one pair of the same tokens for AccessPointInfo item all over the messages.

The DUT returned GetAccessPointInfoListResponse messages for step 10 contains at least one pair of the same tokens for AccessPointInfo item all over the messages.

The DUT returned GetAccessPointInfoListResponse messages for step 15 contains at least one pair of the same tokens for AccessPointInfo item all over the messages.

The DUT returned more AccessPointInfo items in GetAccessPointInfoListResponse message than it was specified in Limit parameter.

The DUT returned more AccessPointInfo items in GetAccessPointInfoListResponse message than MaxLimit.

The DUT did not send valid GetServiceCapabilitiesResponse.

Note: In case MaxLimit is equal to 0, 1 will be used instead for request.

4.2.5 GET ACCESS POINT INFO LIST – NO LIMIT

Test Label: Get Access Point Info List Verification without Limit.

Test Case ID: ACCESSCONTROL-2-1-5

ONVIF Core Specification Coverage: GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: GetAccessPointInfoList

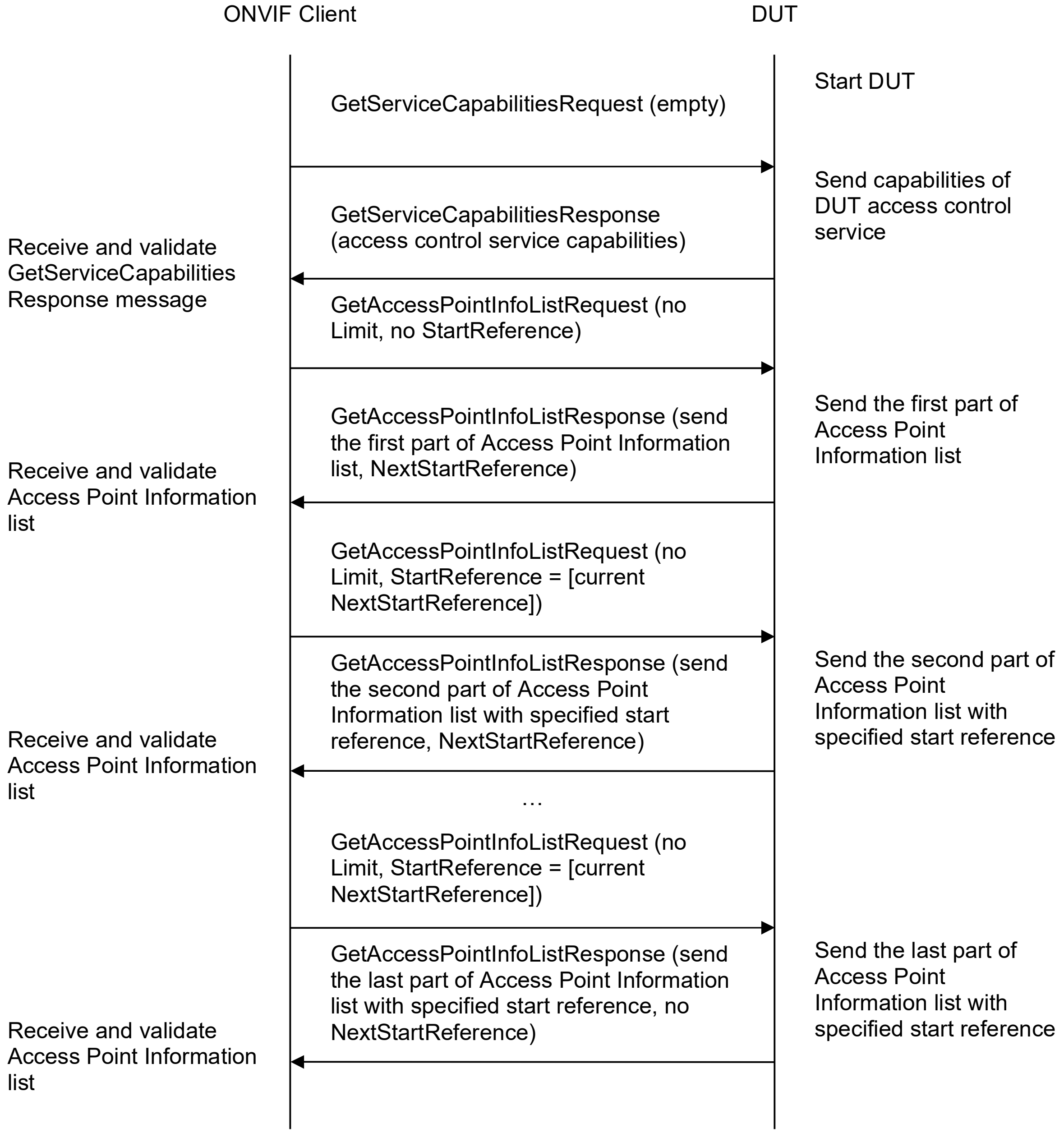
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Get Access Point Info List without using Limit.

Pre-Requisite: Access Control Service was received from the DUT. At least one Access Point is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
GetServiceCapabilitiesRequest (empty)
Send capabilities of DUT access control service
GetServiceCapabilitiesResponse (access control service capabilities)
Receive and validate GetServiceCapabilities Response message
GetAccessPointInfoListRequest (no Limit, no StartReference)
Send the first part of Access Point Information list
GetAccessPointInfoListResponse (send the first part of Access Point Information list, NextStartReference)
Receive and validate Access Point Information list
GetAccessPointInfoListRequest (no Limit, StartReference = [current NextStartReference])
Send the second part of Access Point Information list with specified start reference
GetAccessPointInfoListResponse (send the second part of Access Point Information list with specified start reference, NextStartReference)
Receive and validate Access Point Information list
…
GetAccessPointInfoListRequest (no Limit, StartReference = [current NextStartReference])
Send the last part of Access Point Information list with specified start reference
GetAccessPointInfoListResponse (send the last part of Access Point Information list with specified start reference, no NextStartReference)
Receive and validate Access Point Information list

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve DUT access control service capabilities.

4. Verify the GetServiceCapabilitiesResponse (**MaxLimit**) message from the DUT.

5. ONVIF Client will invoke GetAccessPointInfoListRequest message (no **Limit**, no StartReference) to retrieve the first part of Access Point Information list from the DUT.

6. Verify the GetAccessPointInfoListResponse message from the DUT. Verify that GetAccessPointInfoListResponse message does not contain number of items greater than MaxLimit.

7. If GetAccessPointInfoListResponse message contains NextStartReference, repeat steps 5-6 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting the complete access point list.

8. Verify that GetAccessPointInfoListResponse messages at step 6 does not contain AccessPointInfo items with the same tokens values all over the list, i.e. there shall be no GetAccessPointInfoListResponse messages containing the same value for AccessPointInfo items token.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetAccessPointInfoListResponse message.

The DUT returned GetAccessPointInfoListResponse message that contains wrong item list than requested according to StartReference and Limit values.

The DUT returned GetAccessPointInfoListResponse messages for step 6 contains at least one pair of the same tokens for AccessPointInfo item all over the messages.

The DUT returned more AccessPointInfo items in GetAccessPointInfoListResponse message than MaxLimit.

The DUT did not send valid GetServiceCapabilitiesResponse.

4.2.6 GET ACCESS POINT INFO – TOO MANY ITEMS

Test Label: Get Access Point Info Verification (Too Many Items).

Test Case ID: ACCESSCONTROL-2-1-6

ONVIF Core Specification Coverage: GetAccessPointInfo (ONVIF Access Control Service Specification)

Command under test: GetAccessPointInfo

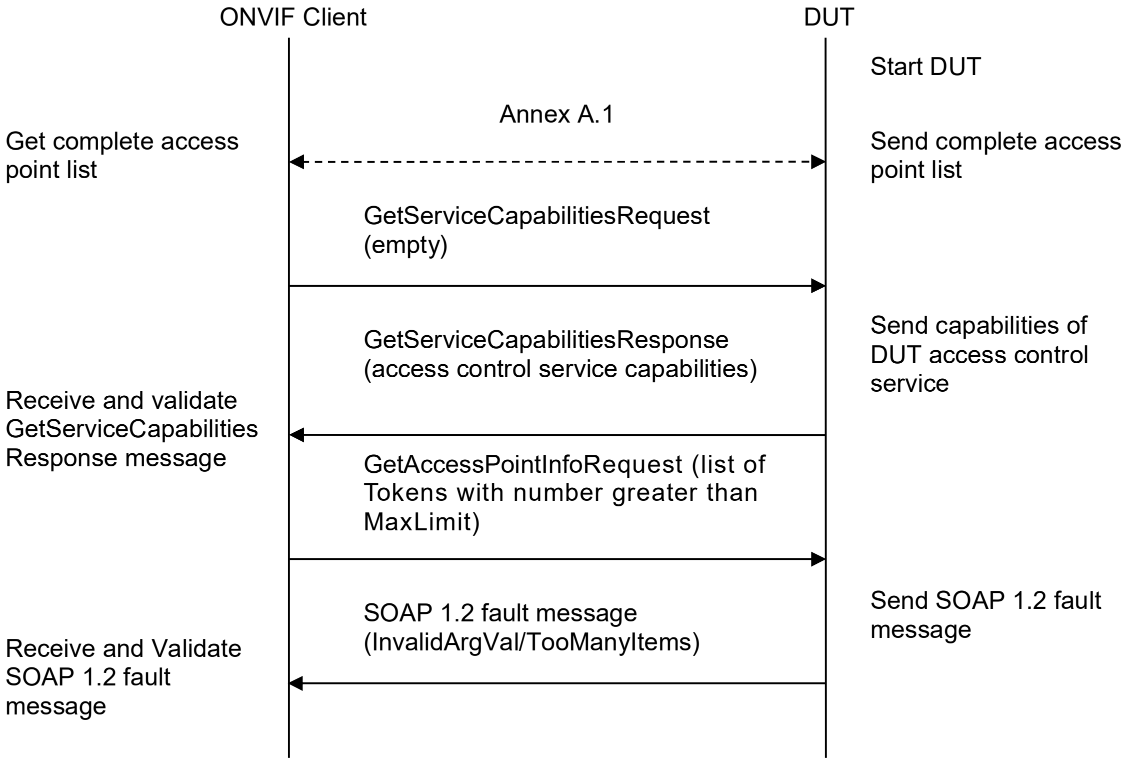
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Get Access Point Info in case if there a more items than MaxLimit in request.

Pre-Requisite: Access Control Service was received from the DUT. At least one Access Point is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetServiceCapabilitiesRequest (empty)
GetServiceCapabilitiesResponse (access control service capabilities)
Send capabilities of DUT access control service
Receive and validate GetServiceCapabilities Response message
GetAccessPointInfoRequest (list of Tokens with number greater than MaxLimit)
SOAP 1.2 fault message (InvalidArgVal/TooManyItems)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get complete list of access points from the DUT (see Annex A.1).
4. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve MaxLimit capability.
5. Verify the GetServiceCapabilitiesResponse message from the DUT.
6. If total number of access points is less than MaxLimit, skip other steps and go to the text test.
7. ONVIF Client will invoke GetAccessPointInfoRequest message (**Token** list with subset of AccessPointInfo.token values from a complete list of access points at step 3 with token

number greater than MaxLimit) to retrieve a subset of Access Point Information from the DUT.

8. The DUT will generate SOAP 1.2 fault message (**InvalidArgs/TooManyItems**).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT sent incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

The DUT did not send valid GetServiceCapabilitiesResponse.

The DUT did not return at least one Access Point on step 3.

Note: If the DUT did not return any AccessPointInfo on complete list of access points at step 3, skip steps 4-8, fail test and go to the next test.

4.2.7 GET ACCESS POINT STATE

Test Label: Get Access Point State Verification.

Test Case ID: ACCESSCONTROL-2-1-7

ONVIF Core Specification Coverage: GetAccessPointState (ONVIF Access Control Service Specification)

Command under test: GetAccessPointState

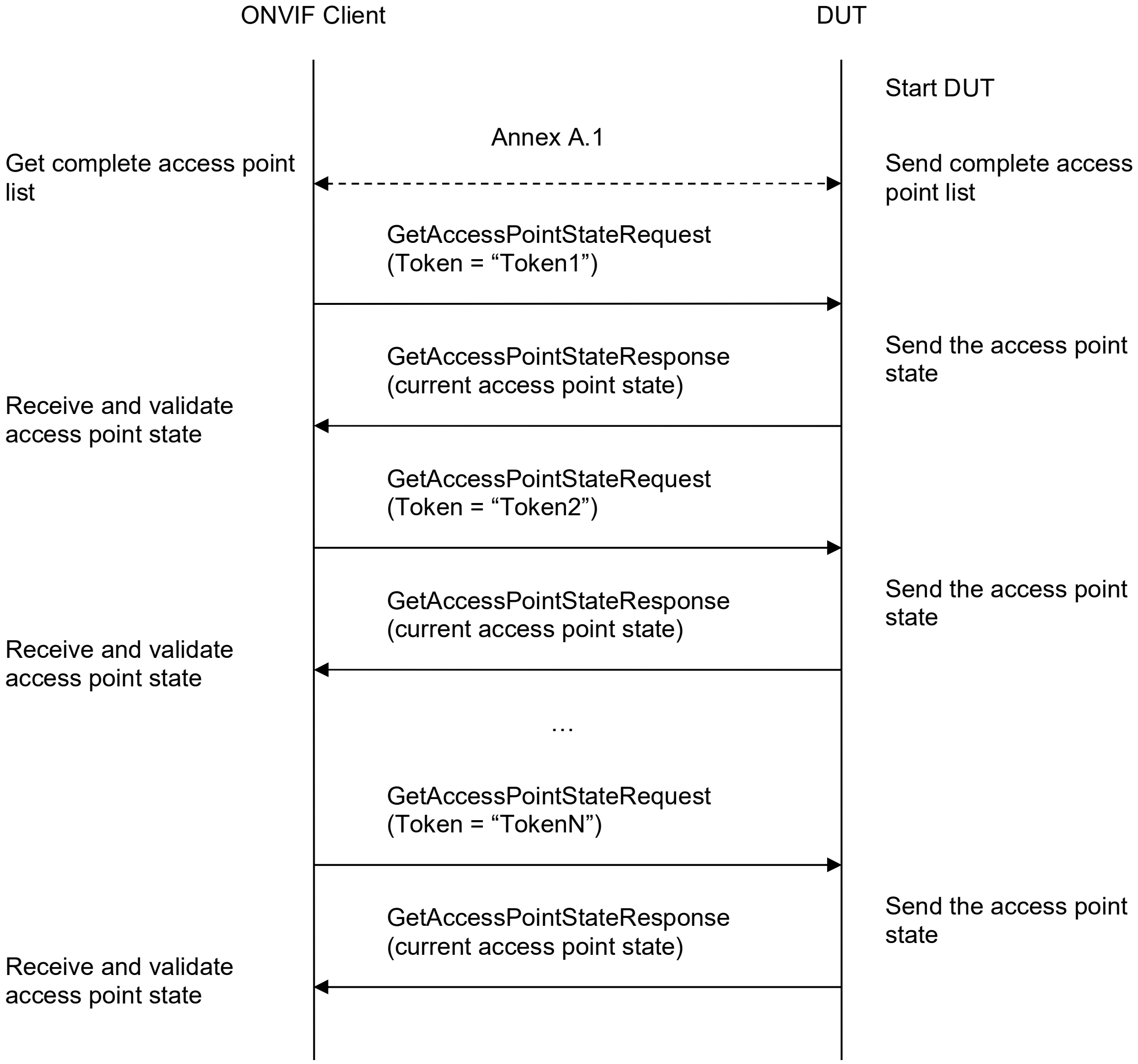
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Get Access Point State.

Pre-Requisite: Access Control Service was received from the DUT. At least one Access Point is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetAccessPointStateRequest (Token = "Token1")
GetAccessPointStateResponse (current access point state)
Send the access point state
Receive and validate access point state
GetAccessPointStateRequest (Token = "Token2")
GetAccessPointStateResponse (current access point state)
Send the access point state
Receive and validate access point state
…
GetAccessPointStateRequest (Token = "TokenN")
GetAccessPointStateResponse (current access point state)
Send the access point state
Receive and validate access point state

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of access points from the DUT (see Annex A.1).
4. ONVIF Client will invoke GetAccessPointStateRequest message (**TokenList.Token = Token1**, where Token1 is the first token from the complete list of access points at step 3) to retrieve Access Point state for specified token from the DUT.
5. Verify the GetAccessPointStateResponse message from the DUT. Check that AccessPointState.Enabled is equal to true, if Access Point does not supports DisableAccessPoint capability, e.g. AccessPointInfo.Capabilities.DisableAccessPoint = false.
6. Repeat steps 4-5 for all other tokens from complete list of access points at step 3.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetAccessPointStateResponse message.

The DUT did not return at least one Access Point on step 3.

The DUT returned AccessPointState.Enabled = false for at least one Access Point in case AccessPointInfo.Capabilities.DisableAccessPoint = false for this Access Point.

Note: If the DUT did not return any AccessPointInfo on complete list of access points at step 3, skip steps 4-6, fail the test and go to the next test.

4.2.8 GET ACCESS POINT STATE WITH INVALID TOKEN

Test Label: Get Access Point State Verification with Invalid Token.

Test Case ID: ACCESSCONTROL-2-1-8

ONVIF Core Specification Coverage: GetAccessPointState (ONVIF Access Control Service Specification)

Command under test: GetAccessPointState

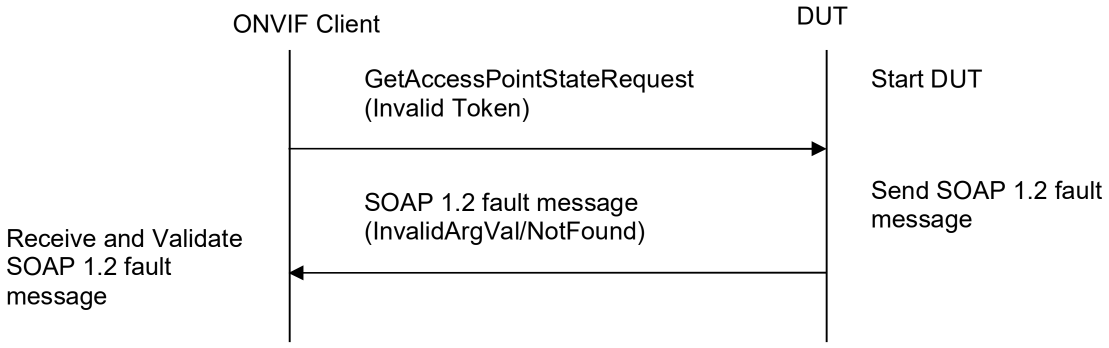
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Get Access Point State with invalid Token.

Pre-Requisite: Access Control Service address was received from the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
GetAccessPointStateRequest (Invalid Token)
Start DUT
SOAP 1.2 fault message (InvalidArgVal/NotFound)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.
3. ONVIF Client will invoke GetAccessPointStateRequest message **(invalid Token**).
4. The DUT will generate SOAP 1.2 fault message (**InvalidArgVal/NotFound**).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT sent incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

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

4.2.9 ENABLE/DISABLE ACCESS POINT

Test Label: Enable and Disable Access Point Verification.

Test Case ID: ACCESSCONTROL-2-1-9

ONVIF Core Specification Coverage: EnableAccessPoint (ONVIF Access Control Service Specification), DisableAccessPoint (ONVIF Access Control Service Specification), GetAccessPointState (ONVIF Access Control Service Specification)

Command under test: EnableAccessPoint, DisableAccessPoint, GetAccessPointState

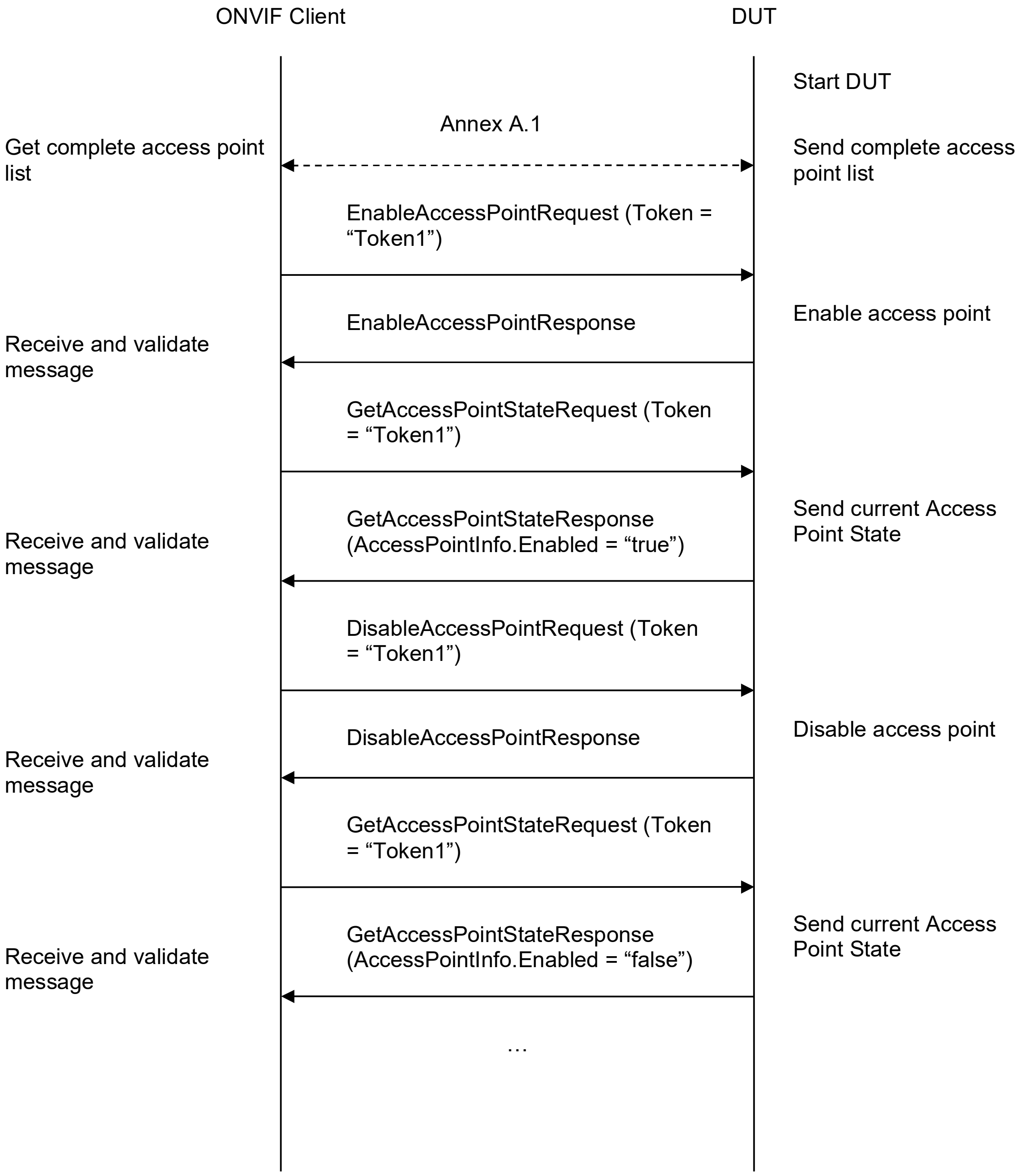
WSDL Reference: accesscontrol.wsdl

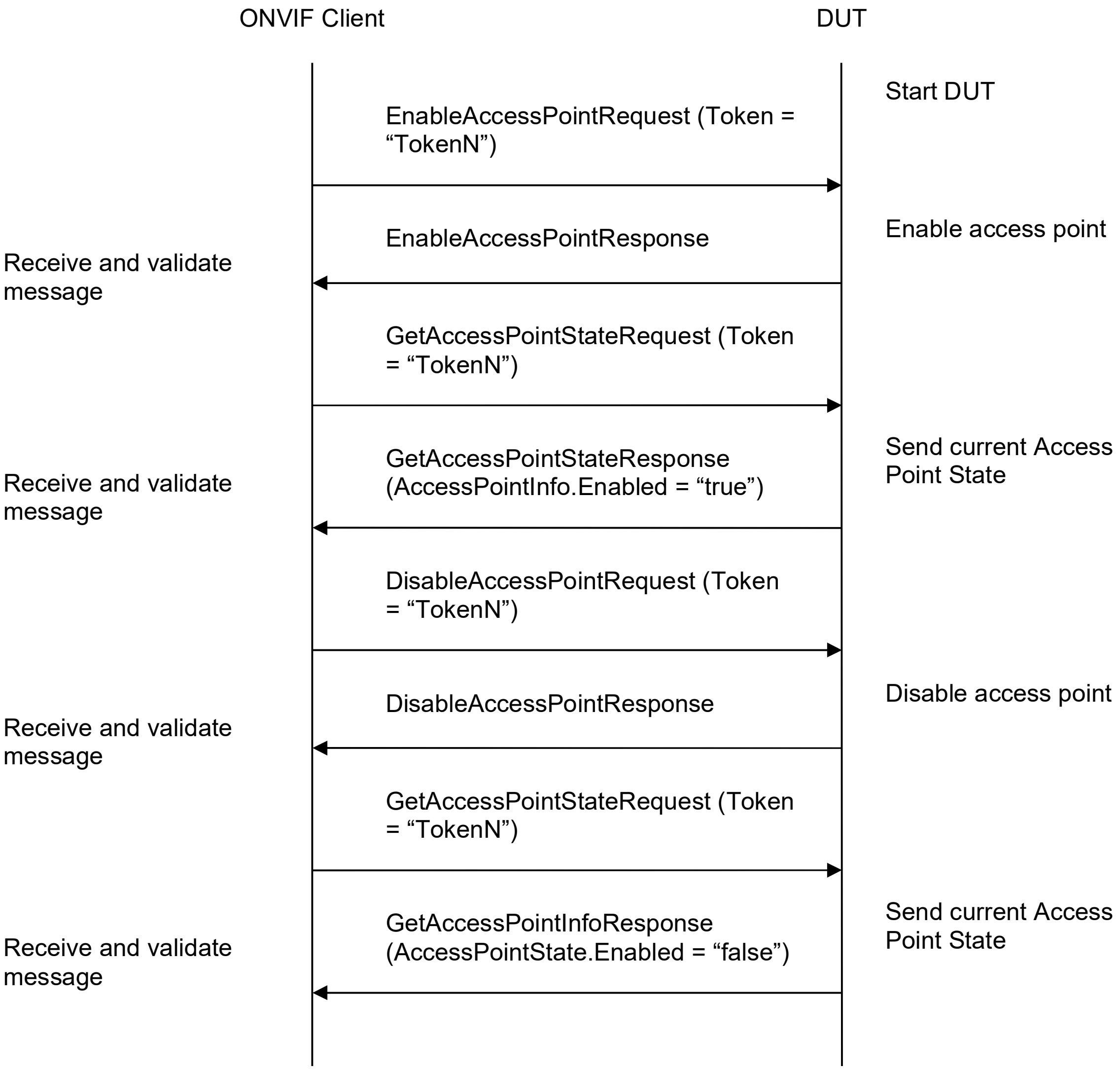
Test Purpose: To verify that Enable and Disable Access Point and its State Change.

Pre-Requisite: Access Control Service was received from the DUT. At least one Access Point with Enable/Disable capability equal to true is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
EnableAccessPointRequest (Token = "Token1")
EnableAccessPointResponse
Enable access point
Receive and validate message
GetAccessPointStateRequest (Token = "Token1")
GetAccessPointStateResponse (AccessPointInfo.Enabled = "true")
Send current Access Point State
Receive and validate message
DisableAccessPointRequest (Token = "Token1")
DisableAccessPointResponse
Disable access point
Receive and validate message
GetAccessPointStateRequest (Token = "Token1")
GetAccessPointStateResponse (AccessPointInfo.Enabled = "false")
Send current Access Point State
Receive and validate message
…


ONVIF Client
DUT
Start DUT
EnableAccessPointRequest (Token = “TokenN”)
EnableAccessPointResponse
Enable access point
Receive and validate message
GetAccessPointStateRequest (Token = “TokenN”)
GetAccessPointStateResponse (AccessPointInfo.Enabled = “true”)
Send current Access Point State
Receive and validate message
DisableAccessPointRequest (Token = “TokenN”)
DisableAccessPointResponse
Disable access point
Receive and validate message
GetAccessPointStateRequest (Token = “TokenN”)
GetAccessPointInfoResponse (AccessPointState.Enabled = “false”)
Send current Access Point State
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get complete list of access points from the DUT (see Annex A.1).
4. If there is no Access Points with **DisableAccessPoint** equal to **true**, fail the test and skip other steps.
5. If Access Point with **Token1** (Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) has **AccessPointInfo.Capabilities.DisableAccessPoint** equal to **false**, then skip steps 6-29 and go to the step 30.

6. ONVIF Client will invoke GetAccessPointStateRequest message (**Token = “Token1”**, where Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) to get current access point state.

7. Verify the GetAccessPointStateResponse message from the DUT.

8. If Access Point with **Token1** (Token1 is the first **AccessPointState.token** from the complete list of access points at step 3) has **AccessPointState.Enabled** equal to **true**, then skip steps 9-19 and go to the step 20.

9. ONVIF Client will invoke EnableAccessPointRequest message (**Token = “Token1”**, where Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) to try enabling access point.

10. Verify the EnableAccessPointResponse message from the DUT.

11. ONVIF Client will invoke GetAccessPointStateRequest message (**Token = “Token1”**, where Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) to get current access point state.

12. Verify the GetAccessPointStateResponse message from the DUT.

13. Verify that AccessPointState.Enabled is equal to true.

14. ONVIF Client will invoke DisableAccessPointRequest message (**Token = “Token1”**, where Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) to try disabling access point.

15. Verify the DisableAccessPointResponse message from the DUT.

16. ONVIF Client will invoke GetAccessPointStateRequest message (**Token = “Token1”**, where Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) to get current access point state.

17. Verify the GetAccessPointStateResponse message from the DUT.

18. Verify that AccessPointState.Enabled is present and equal to false.

19. Go to step 30.

20. ONVIF Client will invoke DisableAccessPointRequest message (**Token = “Token1”**, where Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) to try disabling access point.

21. Verify the DisableAccessPointResponse message from the DUT.

22. ONVIF Client will invoke GetAccessPointStateRequest message (**Token = “Token1”**, where Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) to get current access point state.

23. Verify the GetAccessPointStateResponse message from the DUT.

24. Verify that AccessPointState.Enabled is present and equal to false.

25. ONVIF Client will invoke EnableAccessPointRequest message (**Token = “Token1”**, where Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) to try enabling access point.

26. Verify the EnableAccessPointResponse message from the DUT.

27. ONVIF Client will invoke GetAccessPointStateRequest message (**Token = "Token1"**, where Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) to get current access point state.

28. Verify the GetAccessPointStateResponse message from the DUT.

29. Verify that AccessPointState.Enabled is equal to true or skipped.

30. Repeat steps 5-29 for all other tokens from the complete list of access points at step 3.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetAccessPointStateResponse message.

The DUT did not send valid EnableAccessPointResponse message.

The DUT did not send valid DisableAccessPointResponse message.

The DUT did not change Access Point Enabled value to false after DisableAccessPointResponse message.

The DUT did not change Access Point Enabled value to true after EnableAccessPointResponse message (Enabled element could be skipped in GetAccessPointStateResponse message for this case).

The DUT did not return at least one Access Point at step 3.

Note: If the DUT did not return any AccessPointInfo on complete list of access points at step 3, skip steps 5-30, fail the test and go to the next test.

4.2.10 ENABLE/DISABLE ACCESS POINT - COMMAND NOT SUPPORTED

Test Label: Enable and Disable Access Point Verification – Command not Supported.

Test Case ID: ACCESSCONTROL-2-1-10

ONVIF Core Specification Coverage: EnableAccessPoint (ONVIF Access Control Service Specification), DisableAccessPoint (ONVIF Access Control Service Specification)

Command under test: EnableAccessPoint, DisableAccessPoint

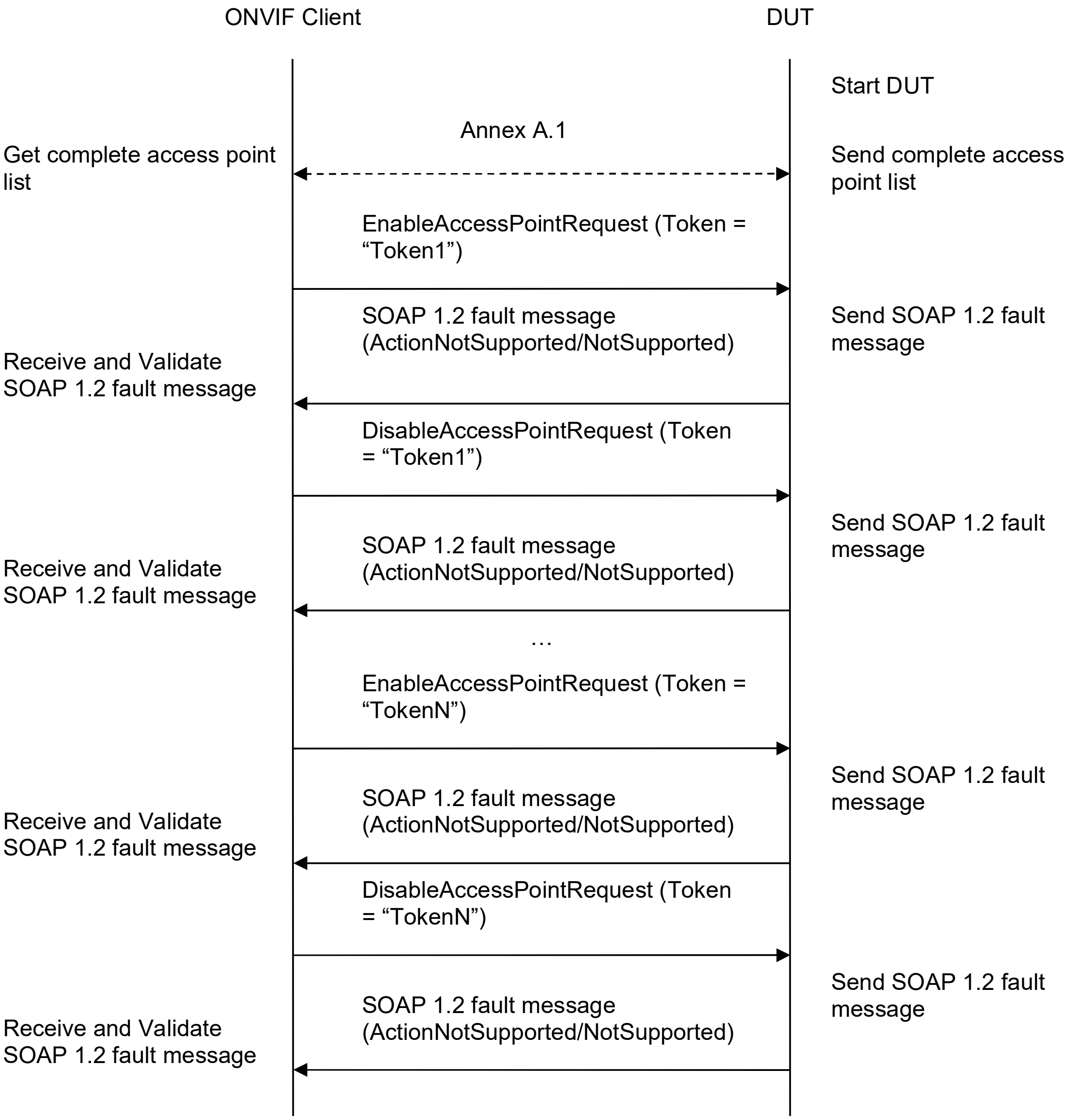
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify that Enable and Disable Access Point in case Door does not support this capability.

Pre-Requisite: Access Control Service was received from the DUT. At least one Access Point is configured and added to the DUT. At least one Access Point with Enable/Disable capability equal to false is configured and added to the DUT, if possible.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
EnableAccessPointRequest (Token = "Token1")
SOAP 1.2 fault message (ActionNotSupported/NotSupported)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message
DisableAccessPointRequest (Token = "Token1")
SOAP 1.2 fault message (ActionNotSupported/NotSupported)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message
…
EnableAccessPointRequest (Token = "TokenN")
SOAP 1.2 fault message (ActionNotSupported/NotSupported)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message
DisableAccessPointRequest (Token = "TokenN")
SOAP 1.2 fault message (ActionNotSupported/NotSupported)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. Get a complete list of access points from the DUT (see Annex A.1).

4. If Access Point with **Token1** (Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) has **AccessPointInfo.Capabilities.DisableAccessPoint** equal to **true** or skipped, then skip steps 5-8 and go to the step 9.

5. ONVIF Client will invoke EnableAccessPointRequest message (**Token = "Token1"**, where Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) to try enabling access point.

6. The DUT will generate SOAP 1.2 fault message (**ActionNotSupported/NotSupported**).

7. ONVIF Client will invoke DisableAccessPointRequest message (**Token = "Token1"**, where Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) to try disabling access point.

8. The DUT will generate SOAP 1.2 fault message (**ActionNotSupported/NotSupported**).

9. Repeat steps 4-8 for all other tokens from the complete list of access points at step 3.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT sent incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

The DUT did not return at least one Access Point on step 3.

Note: If the DUT did not return any AccessPointInfo on complete list of access points at step 3, skip steps 4-9, fail the test and go to the next test.

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

4.2.11 ENABLE ACCESS POINT WITH INVALID TOKEN

Test Label: Enable Access Point Verification with Invalid Token.

Test Case ID: ACCESSCONTROL-2-1-11

ONVIF Core Specification Coverage: EnableAccessPoint (ONVIF Access Control Service Specification)

Command under test: EnableAccessPoint

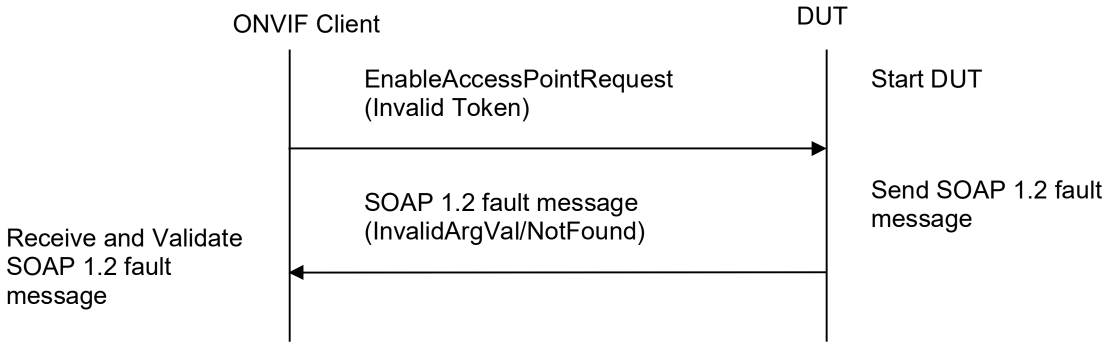
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Enable Access Point with invalid Token.

Pre-Requisite: Access Control Service address was received from the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
EnableAccessPointRequest (Invalid Token)
Start DUT
SOAP 1.2 fault message (InvalidArgVal/NotFound)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke EnableAccessPointRequest message **(invalid Token**).
4. The DUT will generate SOAP 1.2 fault message (**InvalidArgVal/NotFound**).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT sent incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

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

4.2.12 DISABLE ACCESS POINT WITH INVALID TOKEN

Test Label: Disable Access Point Verification with Invalid Token.

Test Case ID: ACCESSCONTROL-2-1-12

ONVIF Core Specification Coverage: DisableAccessPoint (ONVIF Access Control Service Specification)

Command under test: DisableAccessPoint

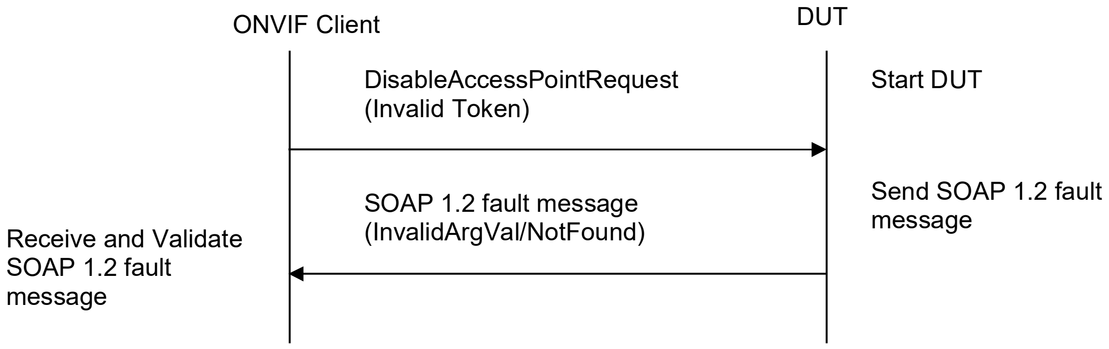
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Disable Access Point with invalid Token.

Pre-Requisite: Access Control Service address was received from the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
DisableAccessPointRequest (Invalid Token)
Start DUT
SOAP 1.2 fault message (InvalidArgVal/NotFound)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke DisableAccessPointRequest message **(invalid Token**).
4. The DUT will generate SOAP 1.2 fault message (**InvalidArgVal/NotFound**).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT sent incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

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

4.3 Area

4.3.1 GET AREA INFO

Test Label: Get Area Info List by Token List Verification.

Test Case ID: ACCESSCONTROL-3-1-1

ONVIF Core Specification Coverage: GetAreaInfo (ONVIF Access Control Service Specification)

Command under test: GetAreaInfo

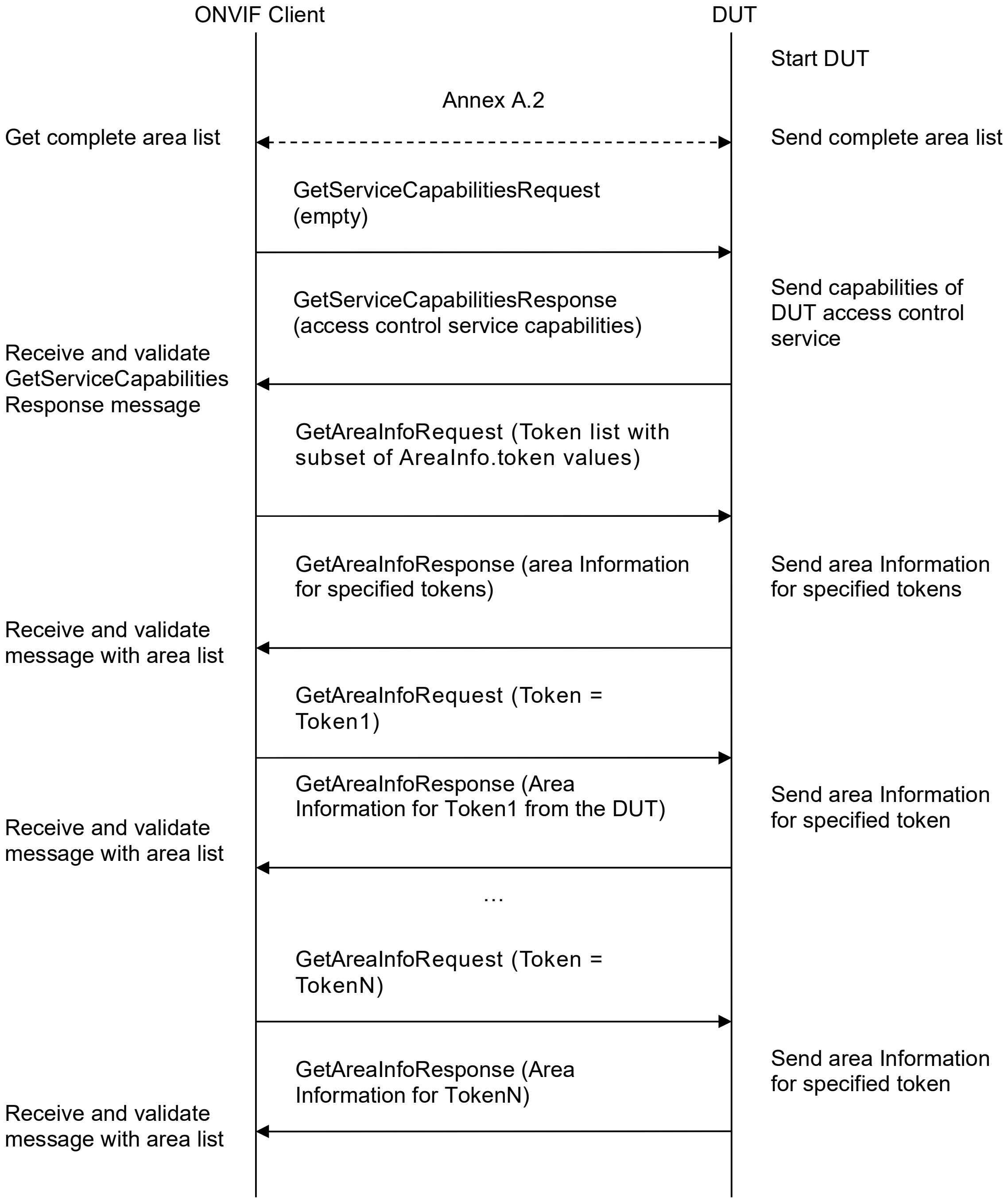
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Get Area Info.

Pre-Requisite: Access Control Service was received from the DUT. At least one Area is configured and added to the DUT, if the DUT allows this.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete area list
Send complete area list
GetServiceCapabilitiesRequest (empty)
GetServiceCapabilitiesResponse (access control service capabilities)
Send capabilities of DUT access control service
Receive and validate GetServiceCapabilities Response message
GetAreaInfoRequest (Token list with subset of AreaInfo.token values)
GetAreaInfoResponse (area Information for specified tokens)
Send area Information for specified tokens
Receive and validate message with area list
GetAreaInfoRequest (Token = Token1)
GetAreaInfoResponse (Area Information for Token1 from the DUT)
Send area Information for specified token
Receive and validate message with area list
…
GetAreaInfoRequest (Token = TokenN)
GetAreaInfoResponse (Area Information for TokenN)
Send area Information for specified token
Receive and validate message with area list

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get the complete list of areas from the DUT (see Annex A.2).
4. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve MaxLimit capability.
5. Verify the GetServiceCapabilitiesResponse message from the DUT.
6. ONVIF Client will invoke GetAreaInfoRequest message (**Token** list with subset of AreaInfo.token values from the complete list of areas at step 3 with token number equal to MaxLimit) to retrieve subset of Area Information from the DUT.
7. Verify the GetAreaInfoResponse message from the DUT.
8. Verify that all requested **AreaInfo** items are in GetAreaInfoResponse message.
9. ONVIF Client will invoke GetAreaInfoRequest message (**Token = Token1**, where Token1 is the first token from the complete list of areas at step 3) to retrieve Area Information for specified token from the DUT.
10. Verify the GetAreaInfoResponse message from the DUT.
11. Verify that GetAreaInfoResponse message contains **AreaInfo** only for specified token.
12. Repeat steps 9-11 for all other tokens from complete list of areas at step 3.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetAreaInfoResponse message.

The DUT returned GetAreaInfoResponse message that contains at least two AreaInfo items with equal tokens.

The DUT did not return requested AreaInfo items in GetAreaInfoResponse message for step 7 or 10.

The DUT returned more items than requested in GetAreaInfoResponse message for step 7 or 10.

The DUT did not send valid GetServiceCapabilitiesResponse.

Note: If the DUT did not return any AreaInfo on complete list of areas at step 3, skip steps 4-12 and go to the next test.

Note: In case MaxLimit is equal to 0, 1 will be used instead in requests.

4.3.2 GET AREA INFO WITH INVALID TOKEN

Test Label: Get Area Info Verification with Invalid Token.

Test Case ID: ACCESSCONTROL-3-1-2

ONVIF Core Specification Coverage: GetAreaInfo (ONVIF Access Control Service Specification)

Command under test: GetAreaInfo

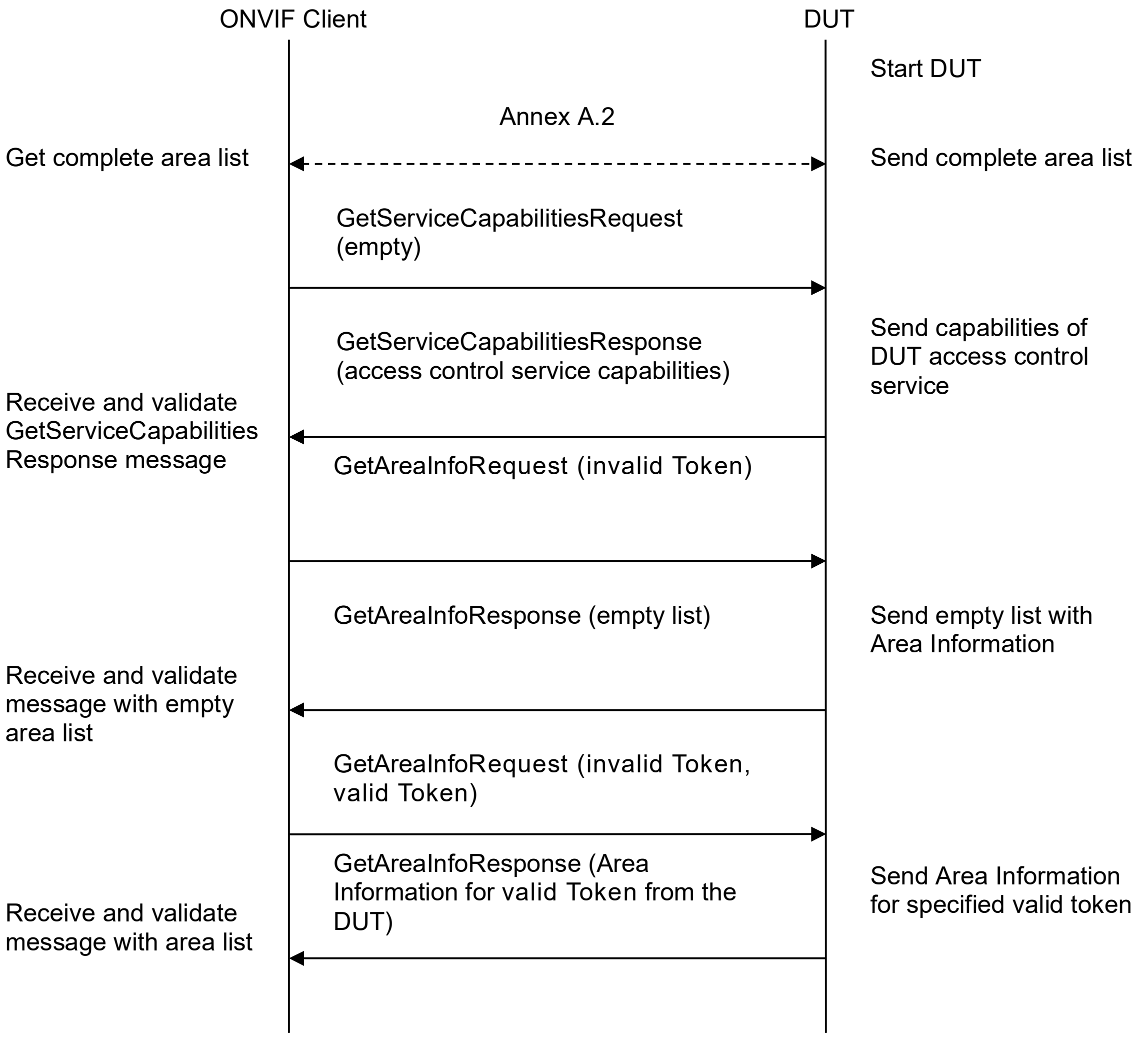
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Get Area Info with invalid Token.

Pre-Requisite: Access Control Service was received from the DUT. At least one Area is configured and added to the DUT, if the DUT allows this.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete area list
Send complete area list
GetServiceCapabilitiesRequest (empty)
GetServiceCapabilitiesResponse (access control service capabilities)
Send capabilities of DUT access control service
Receive and validate GetServiceCapabilities Response message
GetAreaInfoRequest (invalid Token)
GetAreaInfoResponse (empty list)
Send empty list with Area Information
Receive and validate message with empty area list
GetAreaInfoRequest (invalid Token, valid Token)
GetAreaInfoResponse (Area Information for valid Token from the DUT)
Send Area Information for specified valid token
Receive and validate message with area list

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get complete list of access points from the DUT (see Annex A.1).
4. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve MaxLimit capability.
5. Verify the GetServiceCapabilitiesResponse message from the DUT.
6. ONVIF Client will invoke GetAreaInfoRequest message **(invalid Token**).
7. Verify the GetAreaInfoResponse message from the DUT. Check that empty list was returned.
8. ONVIF Client will invoke GetAreaInfoRequest message **(invalid Token, valid Token**).

9. Verify the GetAreaInfoResponse message from the DUT. Check that list which contains Area for valid Token only was returned.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetAreaInfoResponse message.

The DUT returned GetAreaInfoResponse message that contains at least two AreaInfo items with equal tokens.

The DUT did not send valid GetAreaInfoResponse message.

The DUT returned GetAreaInfoResponse message that contains any AreaInfo items for step 7.

The DUT returned GetAreaInfoResponse message that contains any AreaInfo items other than item for valid Token or does not contains AreaInfo item for valid token step 9.

The DUT did not send valid GetServiceCapabilitiesResponse.

Note: If MaxLimit is less than 2, skip steps 8-9.

Note: If the DUT did not return any AreaInfo on complete list of areas at step 3, skip steps 8-9 and go to the next test.

4.3.3 GET AREA INFO LIST – LIMIT

Test Label: Get Area Info List Verification with Limit.

Test Case ID: ACCESSCONTROL-3-1-4

ONVIF Core Specification Coverage: GetAreaInfoList (ONVIF Access Control Service Specification)

Command under test: GetAreaInfoList

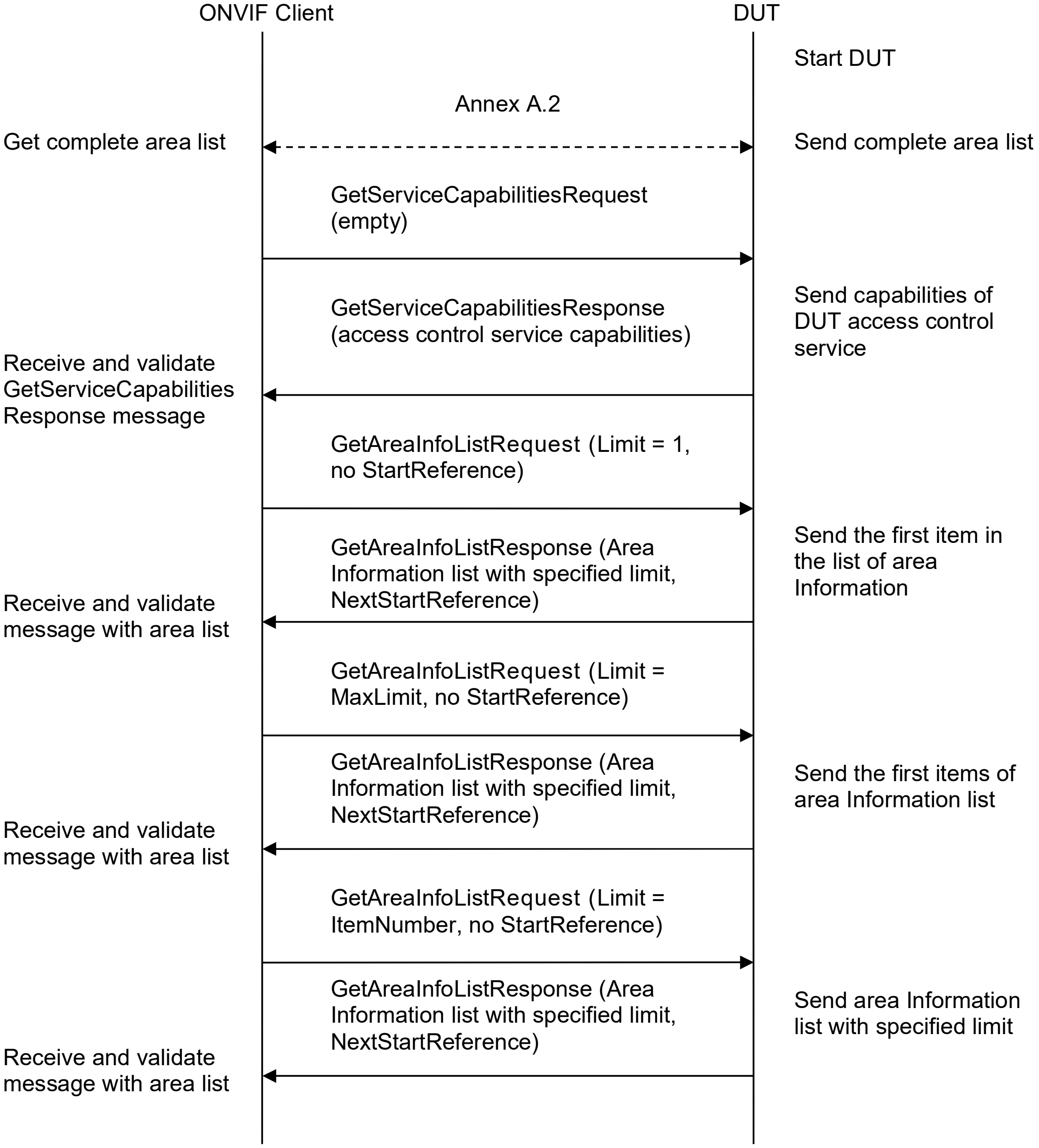
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Get Area Info List using Limit.

Pre-Requisite: Access Control Service was received from the DUT. At least one Area is configured and added to the DUT, if the DUT allows this.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete area list
Send complete area list
GetServiceCapabilitiesRequest (empty)
GetServiceCapabilitiesResponse (access control service capabilities)
Send capabilities of DUT access control service
Receive and validate GetServiceCapabilities Response message
GetAreaInfoListRequest (Limit = 1, no StartReference)
GetAreaInfoListResponse (Area Information list with specified limit, NextStartReference)
Send the first item in the list of area Information
Receive and validate message with area list
GetAreaInfoListRequest (Limit = MaxLimit, no StartReference)
GetAreaInfoListResponse (Area Information list with specified limit, NextStartReference)
Send the first items of area Information list
Receive and validate message with area list
GetAreaInfoListRequest (Limit = ItemNumber, no StartReference)
GetAreaInfoListResponse (Area Information list with specified limit, NextStartReference)
Send area Information list with specified limit
Receive and validate message with area list

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of areas from the DUT (see Annex A.2)
4. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve MaxLimit capability.

5. Verify the GetServiceCapabilitiesResponse message from the DUT.

6. ONVIF Client will invoke GetAreaInfoListRequest message (**Limit = 1, no StartReference**) to retrieve the first item on the list of Area Information from the DUT.

7. Verify the GetAreaInfoListResponse message from the DUT.

8. Verify that GetAreaInfoListResponse message at step 7 does not contain number of items greater than 1. Verify that GetAreaInfoListResponse message at step 7 does not contain number of items greater than MaxLimit.

9. ONVIF Client will invoke GetAreaInfoListRequest message (**Limit = MaxLimit, no StartReference**) to retrieve the first part list of Area Information from the DUT and is limited by MaxLimit.

10. Verify the GetAreaInfoListResponse message from the DUT.

11. Verify that GetAreaInfoListResponse message at step 10 does not contain number of items greater than MaxLimit.

12. ONVIF Client will invoke GetAreaInfoListRequest message (**Limit = ItemNumber, no StartReference,** where ItemNumber between 1 and minimal value between MaxLimit and total number of areas) to retrieve sublist of Area Information, and is limited by ItemNumber from the DUT.

13. Verify the GetAreaInfoListResponse message from the DUT.

14. Verify that GetAreaInfoListResponse message at step 13 does not contain number of items greater than ItemNumber. Verify that GetAreaInfoListResponse message at step 13 does not contain number of items greater than MaxLimit.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetAreaInfoListResponse message.

The DUT returned GetAreaInfoListResponse message that contains wrong item list than requested according to Limit value and StartReference value.

The DUT returned more AreaInfo items in GetAreaInfoListResponse message than it was specified in Limit parameter.

The DUT returned more AreaInfo items in GetAreaInfoListResponse message than MaxLimit.

The DUT did not send valid GetServiceCapabilitiesResponse.

Note: If the DUT did not return any AreaInfo on complete list of areas at step 3, skip steps 4-14 and go to the next test.

Note: In case MaxLimit is equal to 0, 1 will be used instead in requests.

4.3.4 GET AREA INFO LIST – START REFERENCE AND LIMIT

Test Label: Get Area Info List Verification with Start Reference and Limit.

Test Case ID: ACCESSCONTROL-3-1-5

ONVIF Core Specification Coverage: GetAreaInfoList (ONVIF Access Control Service Specification)

Command under test: GetAreaInfoList

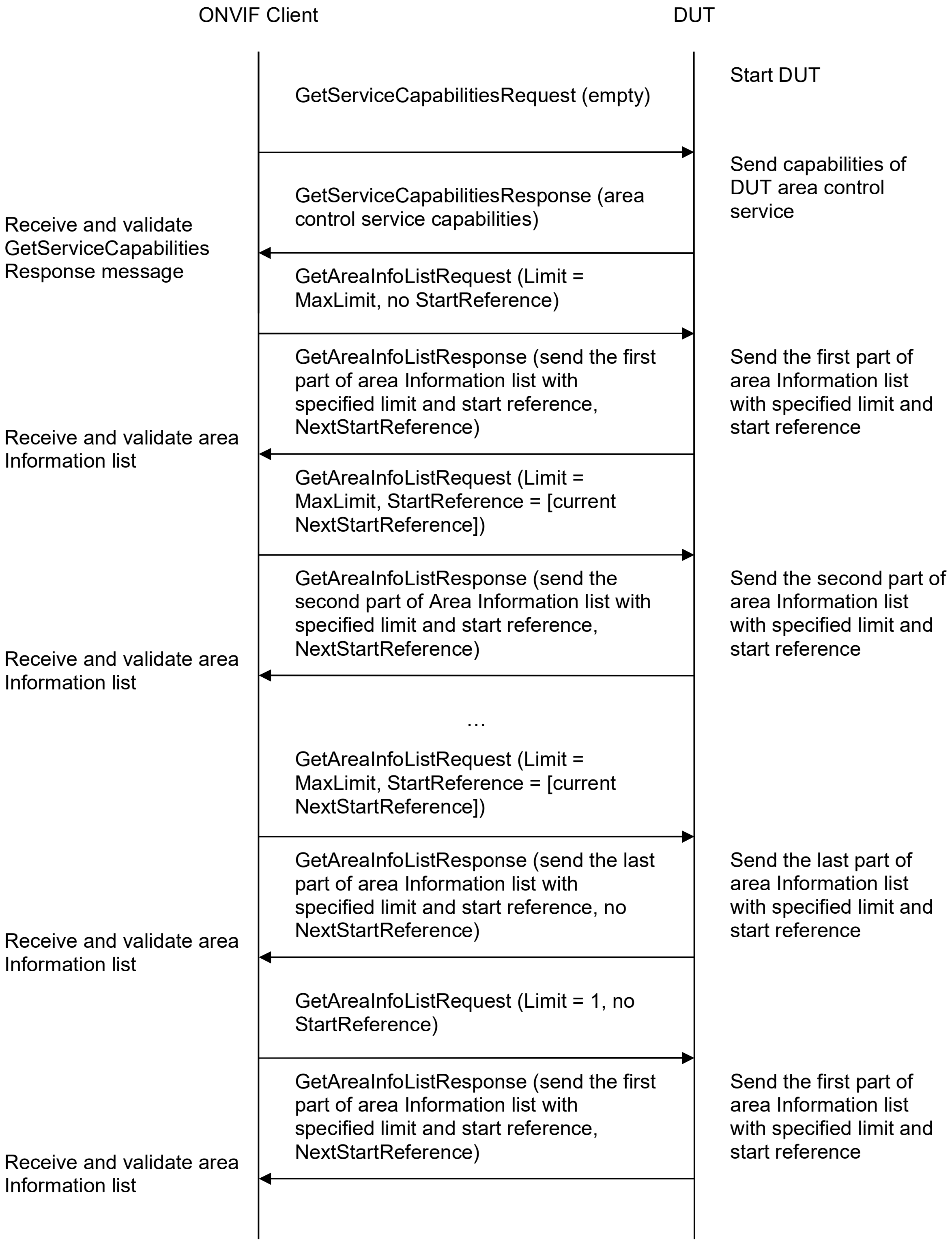
WSDL Reference: accesscontrol.wsdl

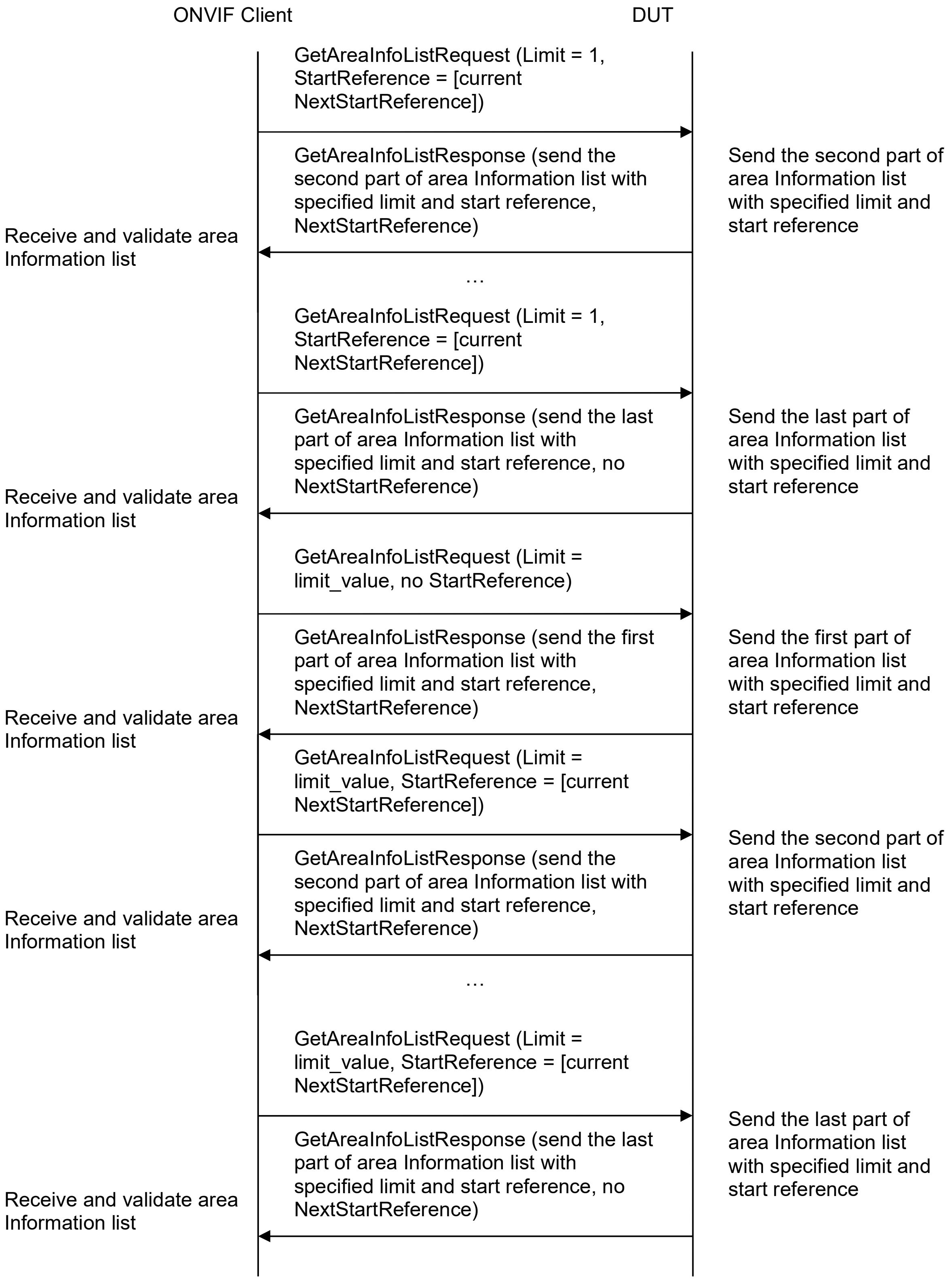
Test Purpose: To verify Get Area Info List using StartReference and Limit.

Pre-Requisite: Access Control Service was received from the DUT. At least one Area is configured and added to the DUT, if the DUT allows this.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
GetServiceCapabilitiesRequest (empty)
Send capabilities of DUT area control service
GetServiceCapabilitiesResponse (area control service capabilities)
Receive and validate GetServiceCapabilities Response message
GetAreaInfoListRequest (Limit = MaxLimit, no StartReference)
GetAreaInfoListResponse (send the first part of area Information list with specified limit and start reference, NextStartReference)
Send the first part of area Information list with specified limit and start reference
Receive and validate area Information list
GetAreaInfoListRequest (Limit = MaxLimit, StartReference = [current NextStartReference])
GetAreaInfoListResponse (send the second part of Area Information list with specified limit and start reference, NextStartReference)
Send the second part of area Information list with specified limit and start reference
Receive and validate area Information list
…
GetAreaInfoListRequest (Limit = MaxLimit, StartReference = [current NextStartReference])
GetAreaInfoListResponse (send the last part of area Information list with specified limit and start reference, no NextStartReference)
Send the last part of area Information list with specified limit and start reference
Receive and validate area Information list
GetAreaInfoListRequest (Limit = 1, no StartReference)
GetAreaInfoListResponse (send the first part of area Information list with specified limit and start reference, NextStartReference)
Send the first part of area Information list with specified limit and start reference
Receive and validate area Information list


ONVIF Client
DUT
GetAreaInfoListRequest (Limit = 1, StartReference = [current NextStartReference])
GetAreaInfoListResponse (send the second part of area Information list with specified limit and start reference, NextStartReference)
Send the second part of area Information list with specified limit and start reference
Receive and validate area Information list
…
GetAreaInfoListRequest (Limit = 1, StartReference = [current NextStartReference])
GetAreaInfoListResponse (send the last part of area Information list with specified limit and start reference, no NextStartReference)
Send the last part of area Information list with specified limit and start reference
Receive and validate area Information list
GetAreaInfoListRequest (Limit = limit_value, no StartReference)
GetAreaInfoListResponse (send the first part of area Information list with specified limit and start reference, NextStartReference)
Send the first part of area Information list with specified limit and start reference
Receive and validate area Information list
GetAreaInfoListRequest (Limit = limit_value, StartReference = [current NextStartReference])
Send the second part of area Information list with specified limit and start reference
GetAreaInfoListResponse (send the second part of area Information list with specified limit and start reference, NextStartReference)
Receive and validate area Information list
…
GetAreaInfoListRequest (Limit = limit_value, StartReference = [current NextStartReference])
Send the last part of area Information list with specified limit and start reference
GetAreaInfoListResponse (send the last part of area Information list with specified limit and start reference, no NextStartReference)
Receive and validate area Information list

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve area control service capabilities of the DUT.

4. Verify the GetServiceCapabilitiesResponse (**MaxLimit**) message from the DUT.

5. ONVIF Client will invoke GetAreaInfoListRequest message (**Limit =** MaxLimit, no StartReference) to retrieve the first part of Area Information list from the DUT.

6. Verify the GetAreaInfoListResponse message from the DUT. Verify that GetAreaInfoListResponse message does not contain number of items greater than Limit. Verify that GetAreaInfoListResponse message does not contain number of items greater than MaxLimit.

7. If GetAreaInfoListResponse message contains NextStartReference, repeat steps 5-6 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting complete area list.

8. Verify that GetAreaInfoListResponse messages at step 6 does not contain AreaInfo items with the same tokens values all over the list, i.e. there shall be no GetAreaInfoListResponse messages containing the same value for AreaInfo items token.

9. ONVIF Client will invoke GetAreaInfoListRequest message (**Limit = 1**, no StartReference) to retrieve the first part of Area Information list from the DUT.

10. Verify the GetAreaInfoListResponse message from the DUT. Verify that GetAreaInfoListResponse message does not contain number of items greater than Limit. Verify that GetAreaInfoListResponse message does not contain number of items greater than MaxLimit.

11. If GetAreaInfoListResponse message contains NextStartReference, repeat steps 5-6 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting complete area list.

12. Verify that GetAreaInfoListResponse messages at step 10 does not contain AreaInfo items with the same tokens values all over the list, i.e. there shall be no GetAreaInfoListResponse messages containing the same value for AreaInfo items token.

13. Verify that total list received during steps 5-7 has the same items as the list received during steps 9-11.

14. ONVIF Client will invoke GetAreaInfoListRequest message (**Limit = [limit_value]**, no StartReference, where limit_value is between 1 and MaxLimit) to retrieve the first part of Area Information list from the DUT.

15. Verify the GetAreaInfoListResponse message from the DUT. Verify that GetAreaInfoListResponse message does not contain number of items greater than Limit. Verify that GetAreaInfoListResponse message does not contain number of items greater than MaxLimit.

16. If GetAreaInfoListResponse message contains NextStartReference, repeat steps 5-6 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting complete area list.

17. Verify that GetAreaInfoListResponse messages at step 15 does not contain AreaInfo items with the same tokens values all over the list, i.e. there shall be no GetAreaInfoListResponse messages containing the same value for AreaInfo items token.

18. Verify that total list received during steps 5-7 has the same items as the list received during steps 14-16.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetAreaInfoListResponse message.

The DUT returned GetAreaInfoListResponse message that contains wrong item list than requested according to Offset and StartReference values.

The DUT returned GetAreaInfoListResponse messages for step 6 contains at least one pair of the same tokens for AreaInfo item all over the messages.

The DUT returned GetAreaInfoListResponse messages for step 10 contains at least one pair of the same tokens for AreaInfo item all over the messages.

The DUT returned GetAreaInfoListResponse messages for step 15 contains at least one pair of the same tokens for AreaInfo item all over the messages.

The DUT returned more AreaInfo items in GetAreaInfoListResponse message than it was specified in Limit parameter.

The DUT did not send valid GetServiceCapabilitiesResponse.

Note: In case MaxLimit is equal to 0, 1 will be used instead in requests.

4.3.5 GET AREA INFO – TOO MANY ITEMS

Test Label: Get Area Info Verification (Too Many Items).

Test Case ID: ACCESSCONTROL-3-1-9

ONVIF Core Specification Coverage: GetAreaInfo (ONVIF Access Control Service Specification)

Command under test: GetAreaInfo

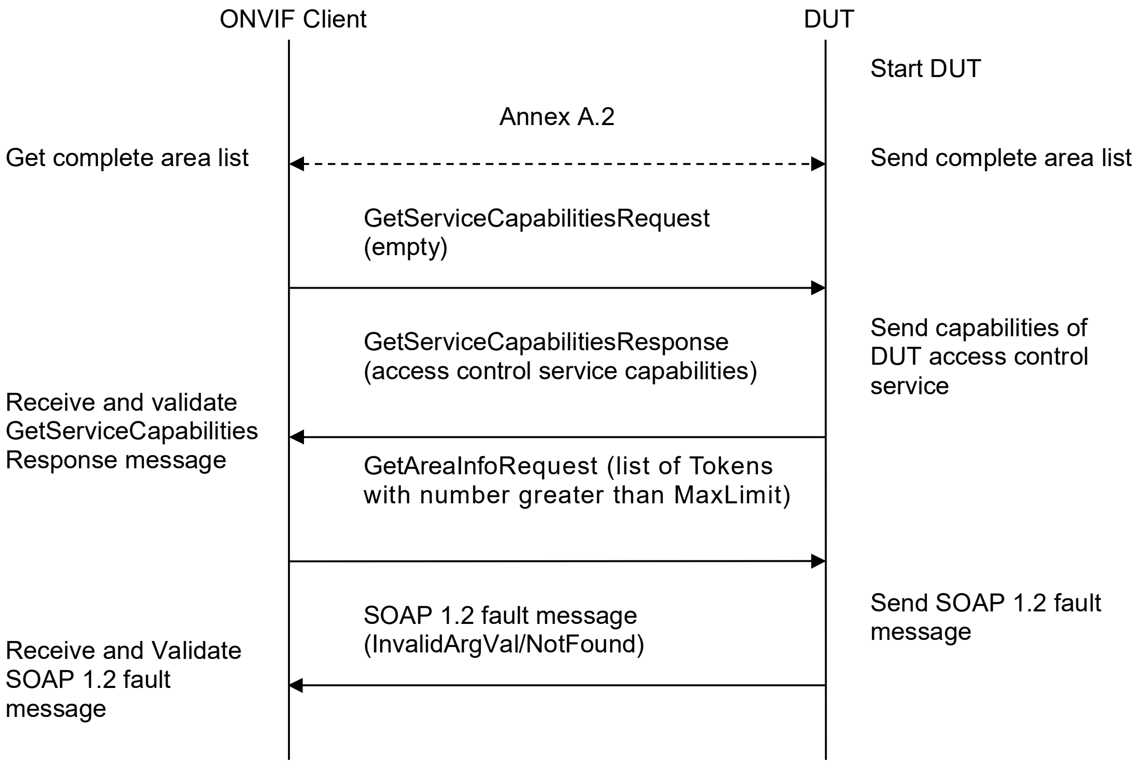
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Get Area Info in case if there a more items than MaxLimit in request.

Pre-Requisite: Access Control Service was received from the DUT. At least one Area is configured and added to the DUT, if the DUT allows this.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete area list
Send complete area list
GetServiceCapabilitiesRequest (empty)
GetServiceCapabilitiesResponse (access control service capabilities)
Send capabilities of DUT access control service
Receive and validate GetServiceCapabilities Response message
GetAreaInfoRequest (list of Tokens with number greater than MaxLimit)
SOAP 1.2 fault message (InvalidArgVal/NotFound)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get the complete list of areas from the DUT (see Annex A.2).
4. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve MaxLimit capability.
5. Verify the GetServiceCapabilitiesResponse message from the DUT.
6. If total number of areas is less than MaxLimit, skip other steps and go to the text test.
7. ONVIF Client will invoke GetAreaInfoRequest message (**Token** list with subset of AreaInfo.token values from a complete list of areas at step 3 with token number greater than MaxLimit) to retrieve a subset of Area Information from the DUT.
8. The DUT will generate SOAP 1.2 fault message (**InvalidArgs/TooManyItems**).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT sent incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

The DUT did not send valid GetServiceCapabilitiesResponse.

Note: If the DUT did not return any AreaInfo on complete list of areas at step 3, skip steps 4-8 and go to the next test.

4.3.6 GET AREA INFO LIST – NO LIMIT

Test Label: Get Area Info List Verification without Limit.

Test Case ID: ACCESSCONTROL-3-1-10

ONVIF Core Specification Coverage: GetAreaInfoList (ONVIF Access Control Service Specification)

Command under test: GetAreaInfoList

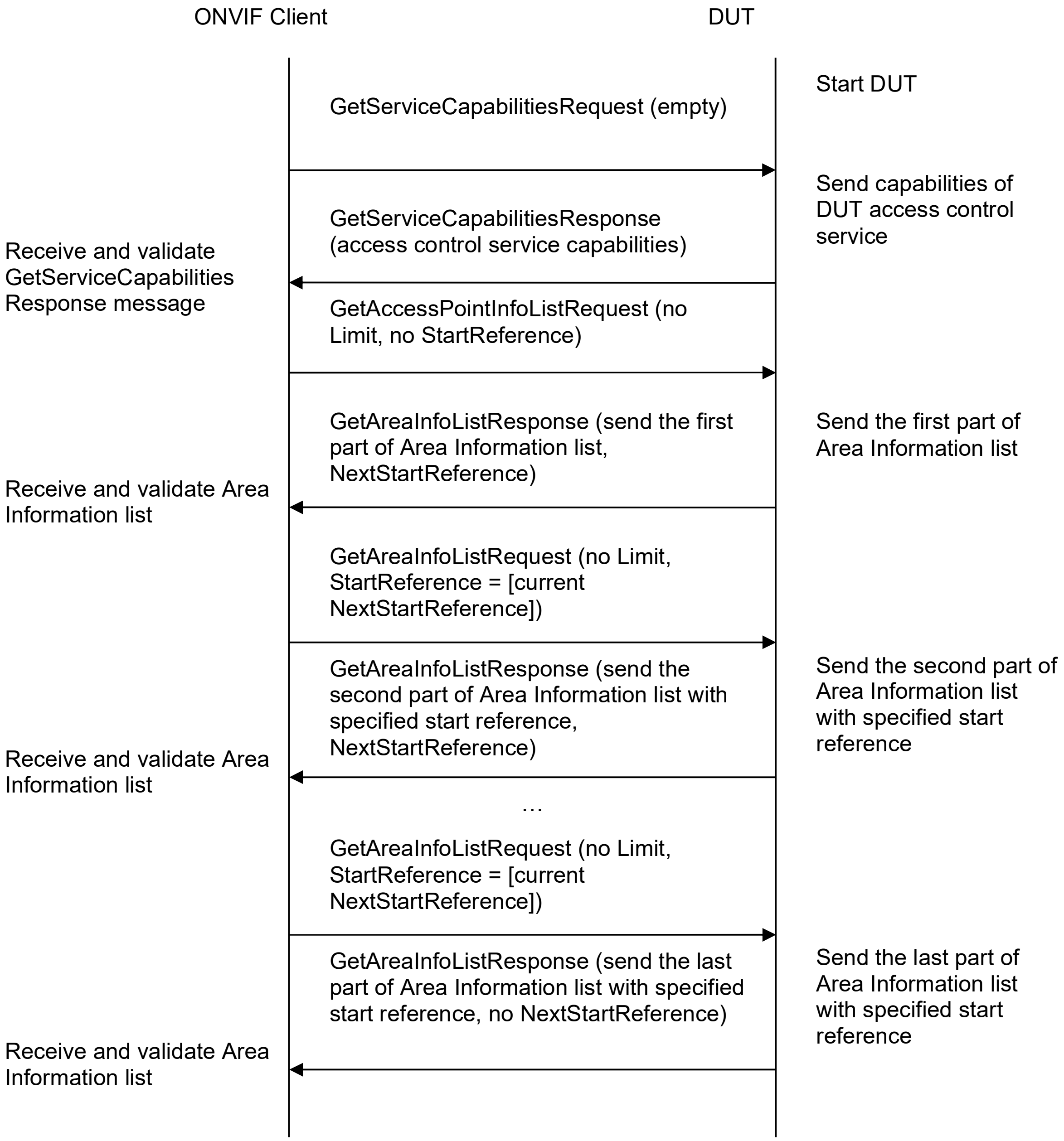
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify Get Area Info List without using Limit.

Pre-Requisite: Access Control Service was received from the DUT. At least one Area is configured and added to the DUT, if the DUT allows this.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
GetServiceCapabilitiesRequest (empty)
Send capabilities of DUT access control service
GetServiceCapabilitiesResponse (access control service capabilities)
Receive and validate GetServiceCapabilities Response message
GetAccessPointInfoListRequest (no Limit, no StartReference)
GetAreaInfoListResponse (send the first part of Area Information list, NextStartReference)
Send the first part of Area Information list
Receive and validate Area Information list
GetAreaInfoListRequest (no Limit, StartReference = [current NextStartReference])
GetAreaInfoListResponse (send the second part of Area Information list with specified start reference, NextStartReference)
Send the second part of Area Information list with specified start reference
Receive and validate Area Information list
…
GetAreaInfoListRequest (no Limit, StartReference = [current NextStartReference])
GetAreaInfoListResponse (send the last part of Area Information list with specified start reference, no NextStartReference)
Send the last part of Area Information list with specified start reference
Receive and validate Area Information list

Test Procedure:

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

5. ONVIF Client will invoke GetAreaInfoListRequest message (no **Limit**, no StartReference) to retrieve the first part of Area Information list from the DUT.
6. Verify the GetAreaInfoListResponse message from the DUT. Verify that GetAreaInfoListResponse message does not contain number of items greater than MaxLimit.
7. If GetAreaInfoListResponse message contains NextStartReference, repeat steps 5-6 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting the complete area list.
8. Verify that GetAreaInfoListResponse messages at step 6 does not contain AreaInfo items with the same tokens values all over the list, i.e. there shall be no GetAreaInfoListResponse messages containing the same value for AreaInfo items token.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetAreaInfoListResponse message.

The DUT returned GetAreaInfoListResponse message that contains wrong item list than requested according to StartReference and Limit values.

The DUT returned GetAreaInfoListResponse messages for step 6 contains at least one pair of the same tokens for AreaInfo item all over the messages.

The DUT returned more AreaInfo items in GetAreaInfoListResponse message than MaxLimit.

The DUT did not send valid GetServiceCapabilitiesResponse.

4.4 External Authorization

4.4.1 ACCESS CONTROL – ACCESS GRANTED TO ANONYMOUS (EXTERNAL AUTHORIZATION)

Test Label: Access Control Service External Authorization Check (Access Granted to Anonymous).

Test Case ID: ACCESSCONTROL-11-1-1

ONVIF Core Specification Coverage: tns1:AccessControl/AccessGranted/Anonymous (Access Control Service Specification), tns1:AccessControl/Request/Anonymous (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification), ExternalAuthorization (ONVIF Access Control Service Specification)

Command under test: ExternalAuthorization

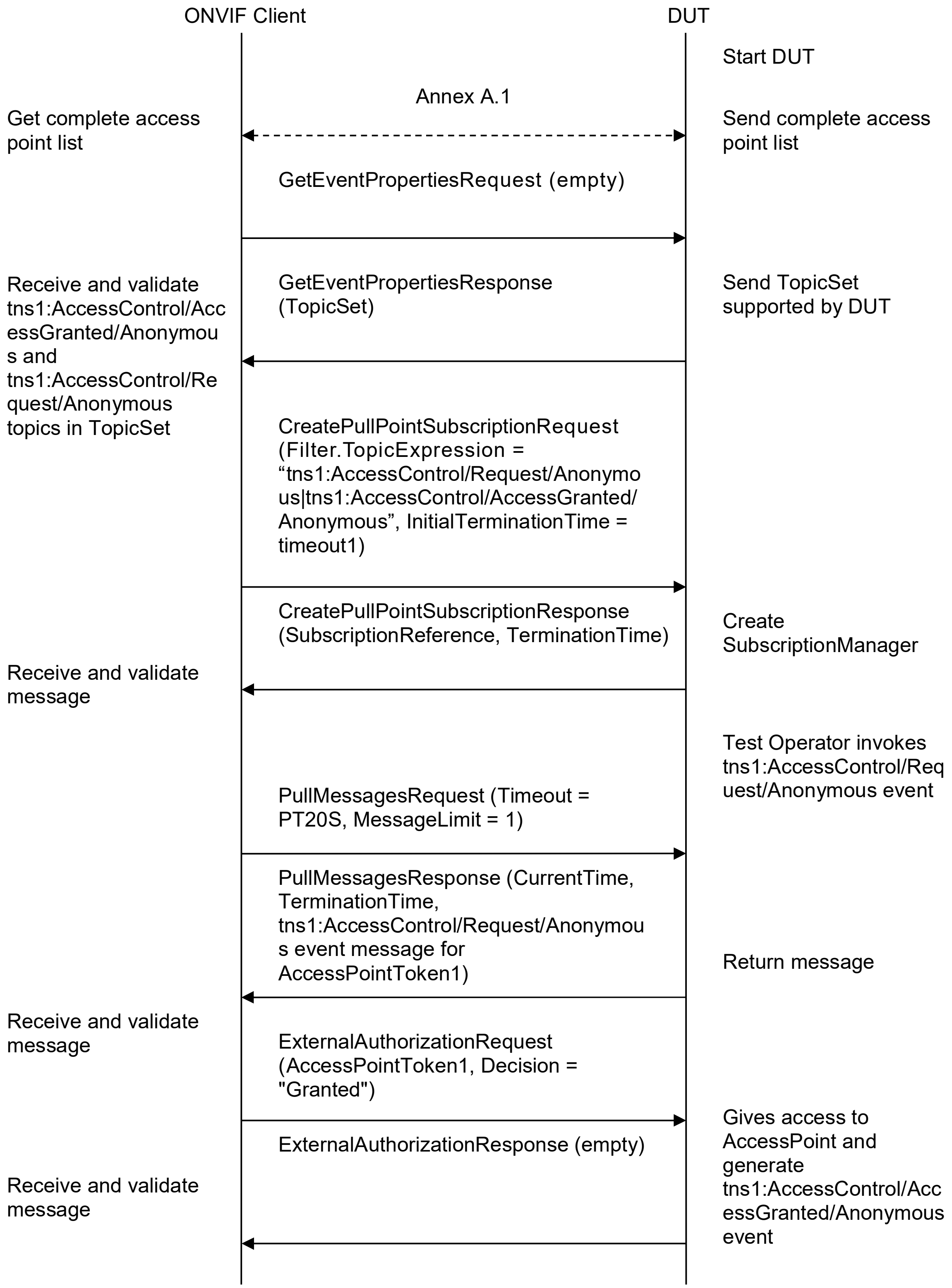
WSDL Reference: event.wsdl, accesscontrol.wsdl

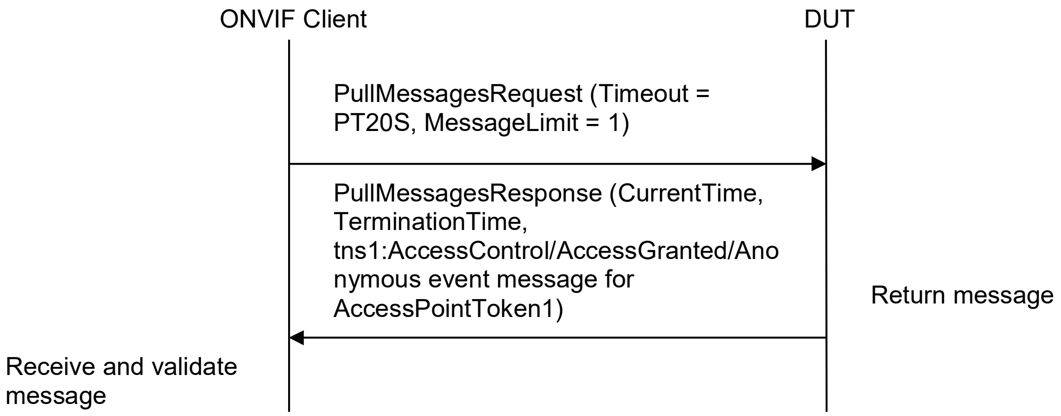
Test Purpose: To verify Access Granted to Anonymous scenario in the case of External Authorization.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with External Authorization capability and Anonymous Access capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/AccessGranted/Anonymous and tns1:AccessControl/Request/Anonymous topics in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/Request/Anonymous|tns1:AccessControl/AccessGranted/Anonymous”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/Request/Anonymous event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Request/Anonymous event message for AccessPointToken1)
Return message
Receive and validate message
ExternalAuthorizationRequest (AccessPointToken1, Decision = "Granted")
Gives access to AccessPoint and generate tns1:AccessControl/AccessGranted/Anonymous event
ExternalAuthorizationResponse (empty)
Receive and validate message


ONVIF Client
DUT
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/AccessGranted/Anonymous event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. Get complete list of access points from the DUT (see Annex A.1).

4. If there are no Access Points with Capabilities.AnonymousAccess = "true" and Capabilities.ExternalAuthorization = "true" in complete list of access points, fail the test and skip other steps.

5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

6. Verify the GetEventPropertiesResponse message from the DUT.

7. Check if there is an event with Topic tns1:AccessControl/AccessGranted/Anonymous. If there is no event with such Topic skip other steps, fail the test and go to the next test.

8. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

10. Check that this event contains Data.SimpleItemDescription item with Name = "External" and Type = "xs:boolean".

11. Check if there is an event with Topic tns1:AccessControl/Request/Anonymous. If there is no event with such Topic skip other steps, fail the test and go to the next test.

12. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

13. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

14. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/Request/Anonymous Topic and tns1:AccessControl/AccessGranted/Anonymous Topic as Filter and an InitialTerminationTime of timeout1.

15. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

16. Test Operator will invoke tns1:AccessControl/Request/Anonymous event for any Access Points with Capabilities.AnonymousAccess = "true" and Capabilities.ExternalAuthorization = "true".

17. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

18. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 17.

19. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

20. Verify that TopicExpression is equal to tns1:AccessControl/Request/Anonymous for the Notification message.

21. Verify that the Notification message contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to one of existing Access Point Tokens Capabilities.AnonymousAccess = "true" and Capabilities.ExternalAuthorization = "true" (e.g. complete list of access points contains Access Point with the same token and this Access Point has Capabilities.AnonymousAccess = "true" and Capabilities.ExternalAuthorization = "true").

22. ONVIF Client will invoke ExternalAuthorizationRequest message (AccessPointToken from Notification message, no CredentialToken, Reason = "Test Access Granted", Decision = "Granted") to Grant Access for Anonymous.

23. Verify that the DUT sends ExternalAuthorizationResponse message.

24. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

25. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 24.

26. Verify that TopicExpression is equal to tns1:AccessControl/AccessGranted/Anonymous for the Notification message.

27. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

28. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to Access Point Token from tns1:AccessControl/Request/Anonymous notification message.

29. Verify that the notification contains Data.SimpleItem item with Name = "External" and its Value is equal to “true”.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send ExternalAuthorizationResponse message.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/Request/Anonymous event at step 18.

The DUT did not send a PullMessagesResponse message that contains one tns1: tns1:AccessControl/AccessGranted/Anonymous event at step 25.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value, invalid External value for tns1:AccessControl/AccessGranted/Anonymous notification message).

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/AccessGranted/Anonymous in GetEventPropertiesResponse.

The DUT did not return valid Topic tns1:AccessControl/Request/Anonymous in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message is received for step 18 or 25 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.4.2 ACCESS CONTROL – ACCESS DENIED TO ANONYMOUS (EXTERNAL AUTHORIZATION)

Test Label: Access Control Service External Authorization Check (Access Denied to Anonymous).

Test Case ID: ACCESSCONTROL-11-1-2

ONVIF Core Specification Coverage: tns1:AccessControl/Denied/Anonymous (Access Control Service Specification), tns1:AccessControl/Request/Anonymous (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification), ExternalAuthorization (ONVIF Access Control Service Specification)

Command under test: ExternalAuthorization

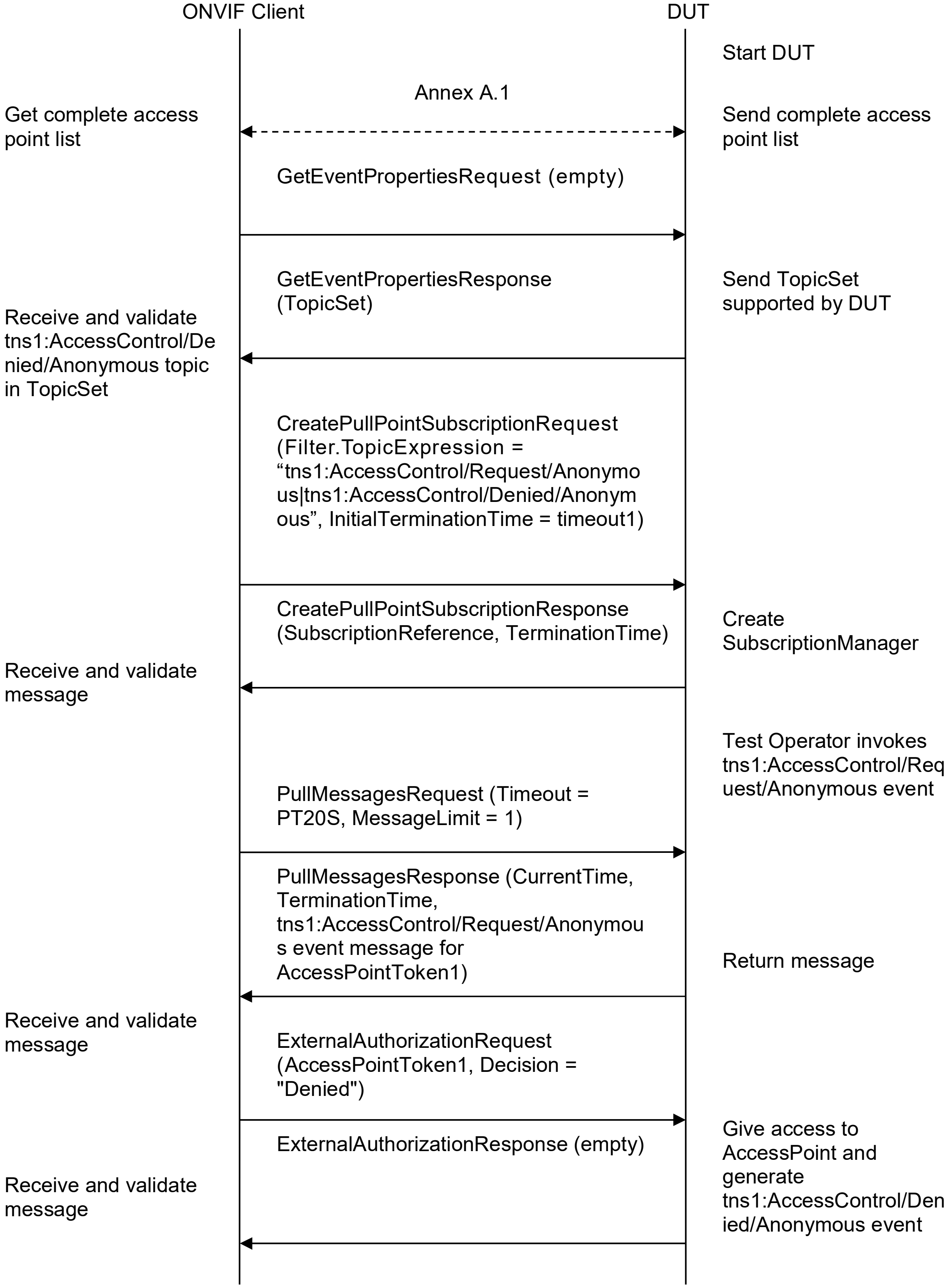
WSDL Reference: event.wsdl, accesscontrol.wsdl

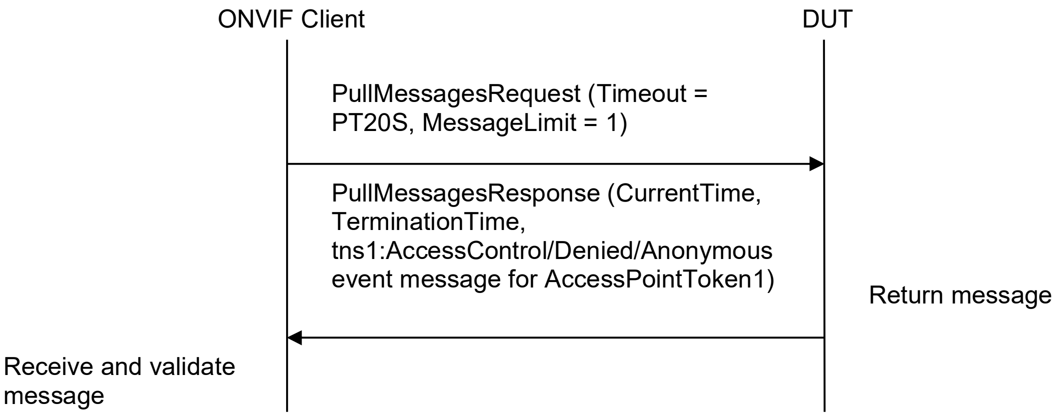
Test Purpose: To verify Access Denied to Anonymous scenario in the case of External Authorization.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with External Authorization capability and Anonymous Access capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/Denied/Anonymous topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/Request/Anonymous|tns1:AccessControl/Denied/Anonymous”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/Request/Anonymous event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Request/Anonymous event message for AccessPointToken1)
Return message
Receive and validate message
ExternalAuthorizationRequest (AccessPointToken1, Decision = "Denied")
ExternalAuthorizationResponse (empty)
Give access to AccessPoint and generate tns1:AccessControl/Denied/Anonymous event
Receive and validate message


ONVIF Client
DUT
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Denied/Anonymous event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. Get complete list of access points from the DUT (see Annex A.1).

4. If there are no Access Points with Capabilities.AnonymousAccess = "true" and Capabilities.ExternalAuthorization = "true" in complete list of access points, fail the test and skip other steps.

5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

6. Verify the GetEventPropertiesResponse message from the DUT.

7. Check if there is an event with Topic tns1:AccessControl/Denied/Anonymous. If there is no event with such Topic skip other steps, fail the test and go to the next test.

8. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

10. Check that this event contains Data.SimpleItemDescription item with Name = "External" and Type = "xs:boolean".

11. Check that this event contains Data.SimpleItemDescription item with Name = "Reason" and Type = " xs:string".

12. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/Request/Anonymous Topic and tns1:AccessControl/Denied/Anonymous Topic as Filter and an InitialTerminationTime of timeout1.

13. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

14. Test Operator will invoke tns1:AccessControl/Request/Anonymous event for any Access Points with Capabilities.AnonymousAccess = "true" and Capabilities.ExternalAuthorization = "true".

15. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

16. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 15.

17. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

18. Verify that TopicExpression is equal to tns1:AccessControl/Request/Anonymous for the Notification message.

19. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to one of existing Access Point Tokens Capabilities.AnonymousAccess = "true" and Capabilities.ExternalAuthorization = "true" (e.g. complete list of access points contains Access Point with the same token and this Access Point has Capabilities.AnonymousAccess = "true" and Capabilities.ExternalAuthorization = "true").

20. ONVIF Client will invoke ExternalAuthorizationRequest message (AccessPointToken from Notification message, no CredentialToken, Reason = "Test Access Denied", Decision = "Denied") to Deny Access for Anonymous.

21. Verify that the DUT sends ExternalAuthorizationResponse message.

22. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

23. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 22.

24. Verify that TopicExpression is equal to tns1:AccessControl/Denied/Anonymous for the Notification message.

25. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

26. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to Access Point Token from tns1:AccessControl/Request/Anonymous notification message.

27. Verify that the notification contains Data.SimpleItem item with Name = "External" and its Value is equal to “true”.

28. Verify that the notification contains Data.SimpleItem item with Name = "Reason" contains Value with type is equal to xs:string. Check that Reason value is one of the following strings: “CredentialNotEnabled”, “CredentialNotActive”, “CredentialExpired”, “InvalidPIN”, “NotPermittedAtThisTime”, “Unauthorized”, “Other”.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send ExternalAuthorizationResponse message.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/Request/Anonymous event at step 16.

The DUT did not send a PullMessagesResponse message that contains one tns1: tns1:AccessControl/Denied/Anonymous event at step 23.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value invalid External or Reason value for tns1:AccessControl/Denied/Anonymous notification message).

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/Denied/Anonymous in GetEventPropertiesResponse.

The DUT did not return valid Topic tns1:AccessControl/Request/Anonymous in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message is received for step 16 or 23 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.4.3 ACCESS CONTROL – ACCESS TIMEOUT TO ANONYMOUS (EXTERNAL AUTHORIZATION)

Test Label: Access Control Service External Authorization Check (Access Timeout to Anonymous).

Test Case ID: ACCESSCONTROL-11-1-3

ONVIF Core Specification Coverage: tns1:AccessControl/Request/Timeout (Access Control Service Specification), tns1:AccessControl/Request/Anonymous (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification), ExternalAuthorization (ONVIF Access Control Service Specification)

Command under test: none

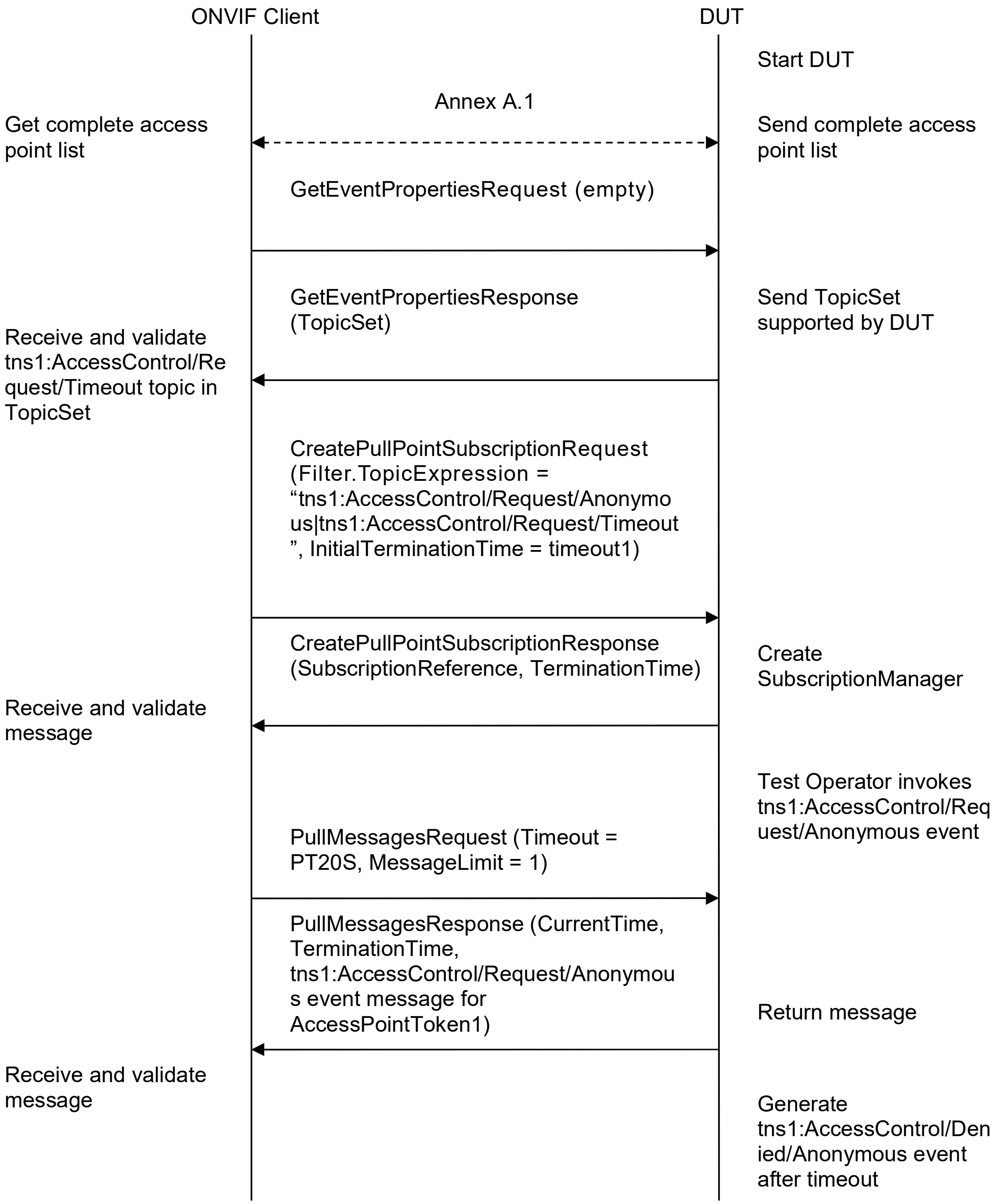
WSDL Reference: event.wsdl, accesscontrol.wsdl

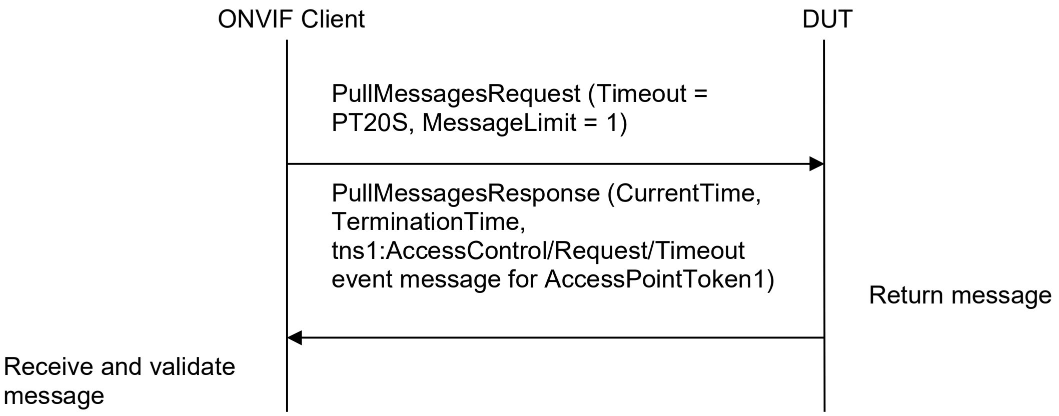
Test Purpose: To verify Access Timeout to Anonymous scenario in the case of External Authorization.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with External Authorization capability and Anonymous Access capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/Request/Timeout topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/Request/Anonymous|tns1:AccessControl/Request/Timeout”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/Request/Anonymous event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Request/Anonymous event message for AccessPointToken1)
Return message
Receive and validate message
Generate tns1:AccessControl/Denied/Anonymous event after timeout


ONVIF Client
DUT
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Request/Timeout event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. Get complete list of access points from the DUT (see Annex A.1).

4. If there are no Access Points with Capabilities.AnonymousAccess = "true" and Capabilities.ExternalAuthorization = "true" in complete list of access points, fail the test and skip other steps.

5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

6. Verify the GetEventPropertiesResponse message from the DUT.

7. Check if there is an event with Topic tns1:AccessControl/Request/Timeout. If there is no event with such Topic skip other steps, fail the test and go to the next test.

8. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

10. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/Request/Anonymous Topic and tns1:AccessControl/Request/Timeout Topic as Filter and an InitialTerminationTime of timeout1.

11. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

12. Test Operator will invoke tns1:AccessControl/Request/Anonymous event for any Access Points with Capabilities.AnonymousAccess = "true" and Capabilities.ExternalAuthorization = "true".

13. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

14. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 13.

15. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

16. Verify that TopicExpression is equal to tns1:AccessControl/Request/Anonymous for the Notification message.

17. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to one of existing Access Point Tokens Capabilities.AnonymousAccess = "true" and Capabilities.ExternalAuthorization = "true" (e.g. complete list of access points contains Access Point with the same token and this Access Point has Capabilities.AnonymousAccess = "true" and Capabilities.ExternalAuthorization = "true").

18. ONVIF Client waits for timeout to initiate tns1:AccessControl/Request/Timeout event.

19. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

20. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 19.

21. Verify that TopicExpression is equal to tns1:AccessControl/Request/Timeout for the Notification message.

22. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

23. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to Access Point Token from tns1:AccessControl/Request/Anonymous notification message.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/Request/Anonymous event at step 14.

The DUT did not send a PullMessagesResponse message that contains one tns1: tns1:AccessControl/Denied/Anonymous event at step 20.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value).

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/Request/Timeout in GetEventPropertiesResponse.

The DUT did not return valid Topic tns1:AccessControl/Request/Anonymous in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message is received for step 16 or 23 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.4.4 ACCESS CONTROL – ACCESS GRANTED WITH CREDENTIAL (EXTERNAL AUTHORIZATION)

Test Label: Access Control Service External Authorization Check (Access Granted with Credential).

Test Case ID: ACCESSCONTROL-11-1-4

ONVIF Core Specification Coverage: tns1:AccessControl/AccessGranted/Credential (Access Control Service Specification), tns1:AccessControl/Request/Credential (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification), ExternalAuthorization (ONVIF Access Control Service Specification)

Command under test: ExternalAuthorization

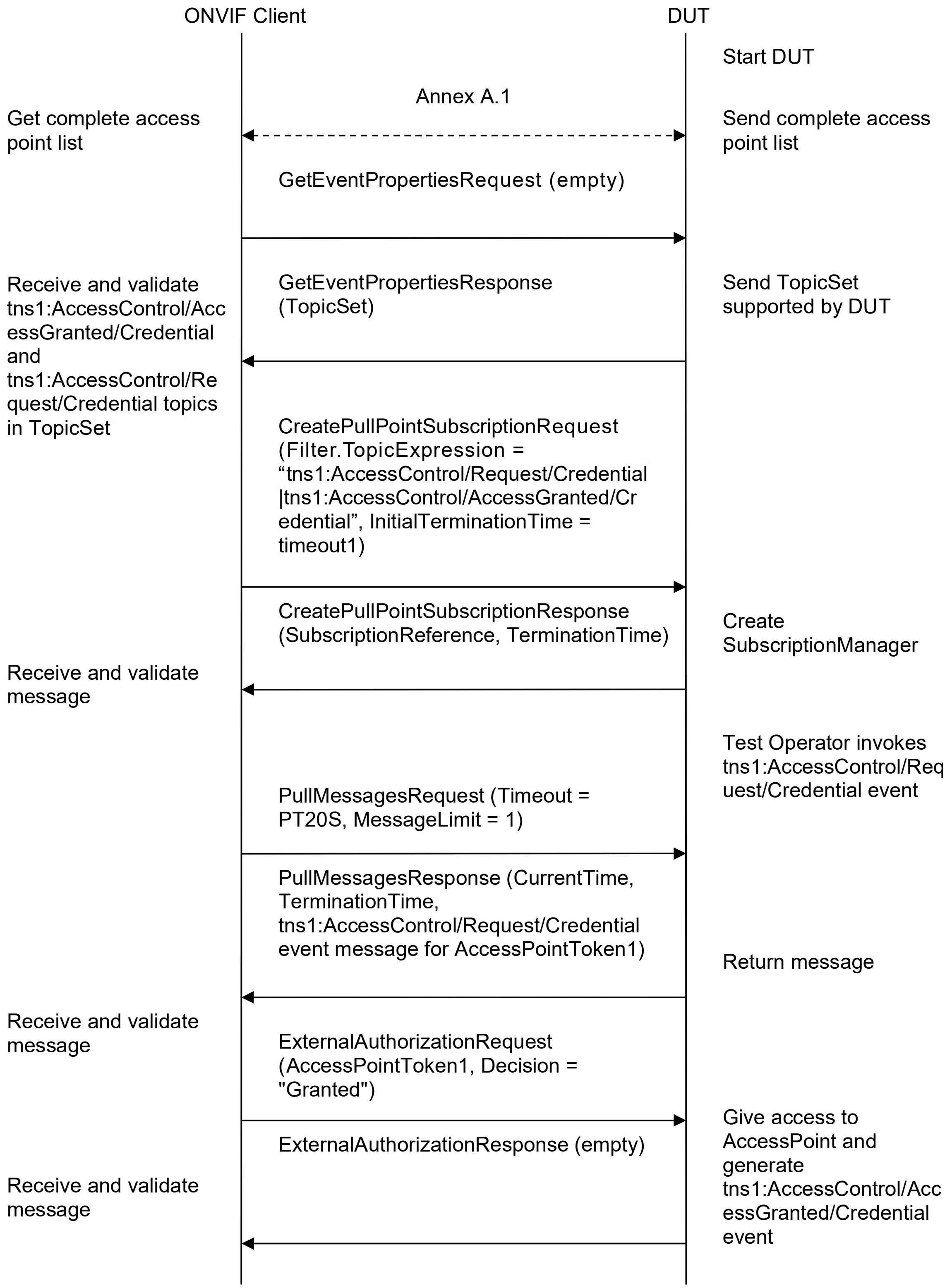
WSDL Reference: event.wsdl, accesscontrol.wsdl

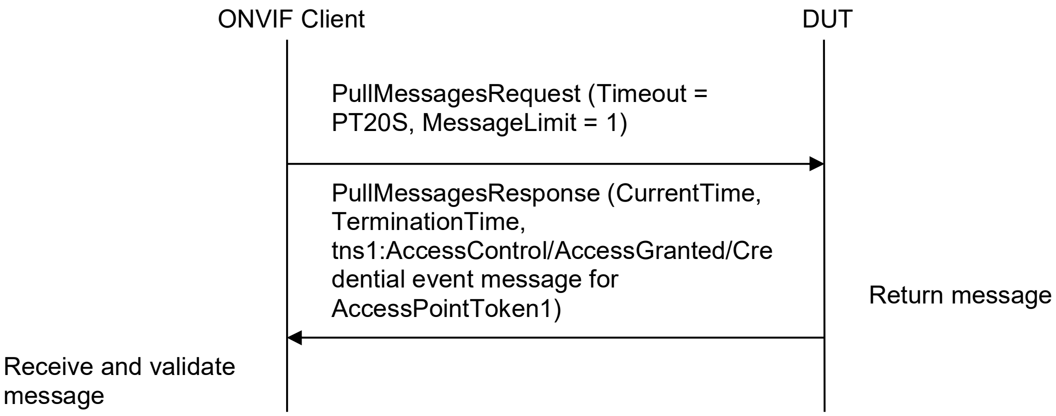
Test Purpose: To verify Access Granted with Credential scenario in the case of External Authorization.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with External Authorization capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/AccessGranted/Credential and tns1:AccessControl/Request/Credential topics in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/Request/Credential |tns1:AccessControl/AccessGranted/Credential”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/Request/Credential event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Request/Credential event message for AccessPointToken1)
Return message
Receive and validate message
ExternalAuthorizationRequest (AccessPointToken1, Decision = "Granted")
Give access to AccessPoint and generate tns1:AccessControl/AccessGranted/Credential event
ExternalAuthorizationResponse (empty)
Receive and validate message


ONVIF Client
DUT
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/AccessGranted/Credential event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. Get complete list of access points from the DUT (see Annex A.1).

4. If there are no Access Point with Capabilities.ExternalAuthorization = "true" in complete list of access points, fail the test and skip other steps.

5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

6. Verify the GetEventPropertiesResponse message from the DUT.

7. Check if there is an event with Topic tns1:AccessControl/AccessGranted/Credential. If there is no event with such Topic skip other steps, fail the test and go to the next test.

8. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

10. Check that this event contains Data.SimpleItemDescription item with Name = "CredentialToken" and Type = "pt:ReferenceToken".

11. Check that if this event contains Data.SimpleItemDescription item with Name = "CredentialHolderName", than it has Type = " xs:string".

12. Check that this event contains Data.SimpleItemDescription item with Name = "External" and Type = "xs:boolean".

13. Check if there is an event with Topic tns1:AccessControl/Request/Credential. If there is no event with such Topic skip other steps, fail the test and go to the next test.

14. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

15. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

16. Check that this event contains Data.SimpleItemDescription item with Name = "CredentialToken" and Type = "pt:ReferenceToken".

17. Check that if this event contains Data.SimpleItemDescription item with Name = "CredentialHolderName", than it has Type = "xs:string".

18. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/Request/Credential Topic and tns1:AccessControl/AccessGranted/Credential Topic as Filter and an InitialTerminationTime of timeout1.

19. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

20. Test Operator will invoke tns1:AccessControl/Request/Credential event for any Access Points with Capabilities.ExternalAuthorization = "true".

21. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

22. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 21.

23. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

24. Verify that TopicExpression is equal to tns1:AccessControl/Request/Credential for the Notification message.

25. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to one of existing Access Point Tokens Capabilities.ExternalAuthorization = "true" (e.g. complete list of access points contains Access Point with the same token and this Access Point has Capabilities.ExternalAuthorization = "true").

26. Verify that the notification contains Data.SimpleItem item with Name = "CredentialToken" and Value with type is equal to pt:ReferenceToken.

27. Verify that the notification which contains Data.SimpleItem item with Name = "CredentialHolderName" contains Value with type is equal to xs:string.

28. ONVIF Client will invoke ExternalAuthorizationRequest message (AccessPointToken from Notification message, CredentialToken from Notification message, Reason = "Test Access Granted", Decision = "Granted") to Grant Access with Credential.

29. Verify that the DUT sends ExternalAuthorizationResponse message.

30. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

31. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 30.

32. Verify that TopicExpression is equal to tns1:AccessControl/AccessGranted/Credential for the Notification message.

33. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

34. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to Access Point Token from tns1:AccessControl/Request/Credential notification message.

35. Verify that the notification contains Data.SimpleItem item with Name = "CredentialToken" and Value with type is equal to pt:ReferenceToken. Verify that this value is equal to the same Data.SimpleItem from tns1:AccessControl/Request/Credential notification.

36. Verify that the notification which contains Data.SimpleItem item with Name = "CredentialHolderName" contains Value with type is equal to xs:string. Verify that this value is equal to the same Data.SimpleItem from tns1:AccessControl/Request/Credential notification.

37. Verify that the notification contains Data.SimpleItem item with Name = "External" and its Value is equal to “true”.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send ExternalAuthorizationResponse message.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/Request/Credential event at step 22.

The DUT did not send a PullMessagesResponse message that contains one tns1: tns1:AccessControl/AccessGranted/Anonymous event at step 31.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value, invalid CredentialToken, invalid CredentialHolderName, invalid External value for tns1:AccessControl/AccessGranted/Credential notification message).

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/AccessGranted/Credential in GetEventPropertiesResponse.

The DUT did not return valid Topic tns1:AccessControl/Request/Credential in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message is received for step 22 or 31 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

PullMessagesFaultResponse.

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.4.5 ACCESS CONTROL – ACCESS DENIED WITH CREDENTIAL (EXTERNAL AUTHORIZATION)

Test Label: Access Control Service External Authorization Check (Access Denied with Credential).

Test Case ID: ACCESSCONTROL-11-1-5

ONVIF Core Specification Coverage: tns1:AccessControl/Denied/Credential (Access Control Service Specification), tns1:AccessControl/Request/Credential (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification), ExternalAuthorization (ONVIF Access Control Service Specification)

Command under test: ExternalAuthorization

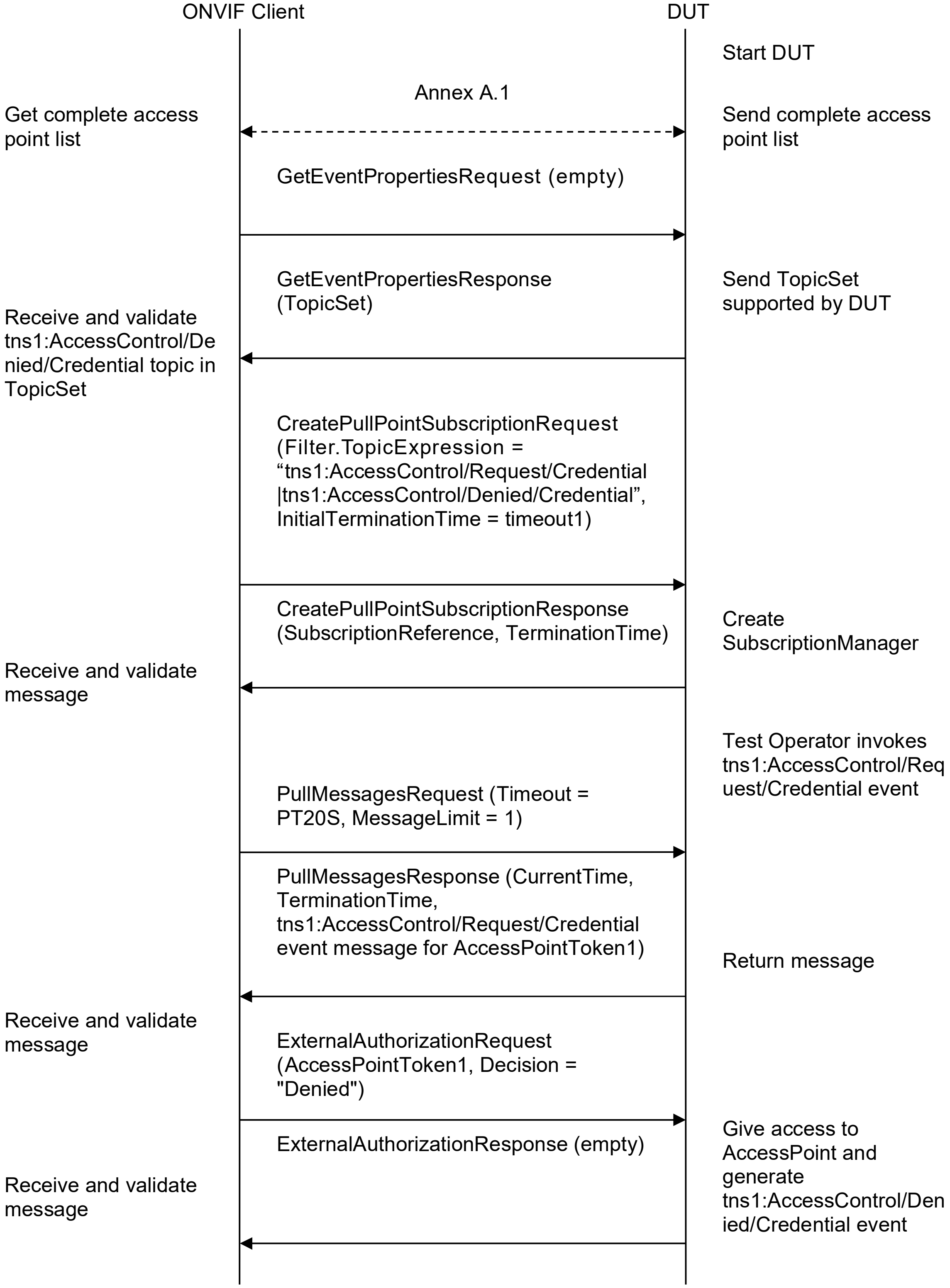
WSDL Reference: event.wsdl, accesscontrol.wsdl

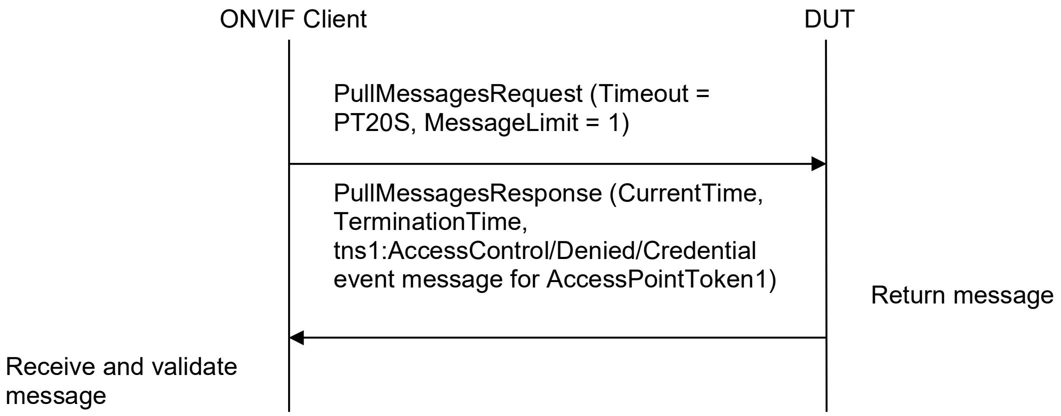
Test Purpose: To verify Access Denied with Credential scenario in the case of External Authorization.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with External Authorization capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/Denied/Credential topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/Request/Credential |tns1:AccessControl/Denied/Credential”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/Request/Credential event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Request/Credential event message for AccessPointToken1)
Return message
Receive and validate message
ExternalAuthorizationRequest (AccessPointToken1, Decision = "Denied")
ExternalAuthorizationResponse (empty)
Give access to AccessPoint and generate tns1:AccessControl/Denied/Credential event
Receive and validate message


ONVIF Client
DUT
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Denied/Credential event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. Get complete list of access points from the DUT (see Annex A.1).

4. If there are no Access Point with and Capabilities.ExternalAuthorization = "true" in complete list of access points, fail the test and skip other steps.

5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

6. Verify the GetEventPropertiesResponse message from the DUT.

7. Check if there is an event with Topic tns1:AccessControl/Denied/Credential. If there is no event with such Topic skip other steps, fail the test and go to the next test.

8. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

10. Check that this event contains Data.SimpleItemDescription item with Name = "CredentialToken" and Type = "pt:ReferenceToken".

11. Check that if this event contains Data.SimpleItemDescription item with Name = "CredentialHolderName", than it has Type = "xs:string".

12. Check that this event contains Data.SimpleItemDescription item with Name = "External" and Type = "xs:boolean".

13. Check that this event contains Data.SimpleItemDescription item with Name = "Reason" and Type = "xs:string".

14. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/Request/Credential Topic and tns1:AccessControl/Denied/Credential Topic as Filter and an InitialTerminationTime of timeout1.

15. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

16. Test Operator will invoke tns1:AccessControl/Request/Credential event for any Access Points with Capabilities.ExternalAuthorization = "true".

17. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

18. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 17.

19. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

20. Verify that TopicExpression is equal to tns1:AccessControl/Request/Credential for the Notification message.

21. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to one of existing Access Point Tokens Capabilities.ExternalAuthorization = "true" (e.g. complete list of access points contains Access Point with the same token and this Access Point has Capabilities.ExternalAuthorization = "true").

22. Verify that the notification contains Data.SimpleItem item with Name = "CredentialToken" and Value with type is equal to pt:ReferenceToken.

23. Verify that the notification which contains Data.SimpleItem item with Name = "CredentialHolderName" contains Value with type is equal to xs:string.

24. ONVIF Client will invoke ExternalAuthorizationRequest message (AccessPointToken from Notification message, CredentialToken from Notification message, Reason = "Test Access Denied", Decision = "Denied") to Deny Access with Credential.

25. Verify that the DUT sends ExternalAuthorizationResponse message.

26. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

27. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 26.

28. Verify that TopicExpression is equal to tns1:AccessControl/Denied/Credential for the Notification message.

29. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

30. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to Access Point Token from tns1:AccessControl/Request/Credential notification message.

31. Verify that the notification contains Data.SimpleItem item with Name = «CredentialToken» and Value with type is equal to pt:ReferenceToken. Verify that this value is equal to the same Data.SimpleItem from tns1:AccessControl/Request/Credential notification.

32. Verify that the notification which contains Data.SimpleItem item with Name = "CredentialHolderName" contains Value with type is equal to xs:string. Verify that this value is equal to the same Data.SimpleItem from tns1:AccessControl/Request/Credential notification.

33. Verify that the notification contains Data.SimpleItem item with Name = "External" and its Value is equal to “true”.

34. Verify that the notification contains Data.SimpleItem item with Name = "Reason" contains Value with type is equal to xs:string. Check that Reason value is one of the following strings: “CredentialNotEnabled”, “CredentialNotActive”, “CredentialExpired”, “InvalidPIN”, “NotPermittedAtThisTime”, “Unauthorized”, “Other”.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send ExternalAuthorizationResponse message.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/Request/Credential event at step 18.

The DUT did not send a PullMessagesResponse message that contains one tns1: tns1:AccessControl/Denied/Anonymous event at step 27.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value, invalid CredentialToken, invalid CredentialHolderName, invalid External value for tns1:AccessControl/Denied/Credential notification message).

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/Denied/Credential in GetEventPropertiesResponse.

The DUT did not return valid Topic tns1:AccessControl/Request/Credential in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message is received for at 18 or 27 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.4.6 ACCESS CONTROL – ACCESS TIMEOUT WITH CREDENTIAL (EXTERNAL AUTHORIZATION)

Test Label: Access Control Service External Authorization Check (Access Timeout with Credential).

Test Case ID: ACCESSCONTROL-11-1-6

ONVIF Core Specification Coverage: tns1:AccessControl/Request/Timeout (Access Control Service Specification), tns1:AccessControl/Request/Credential (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: none

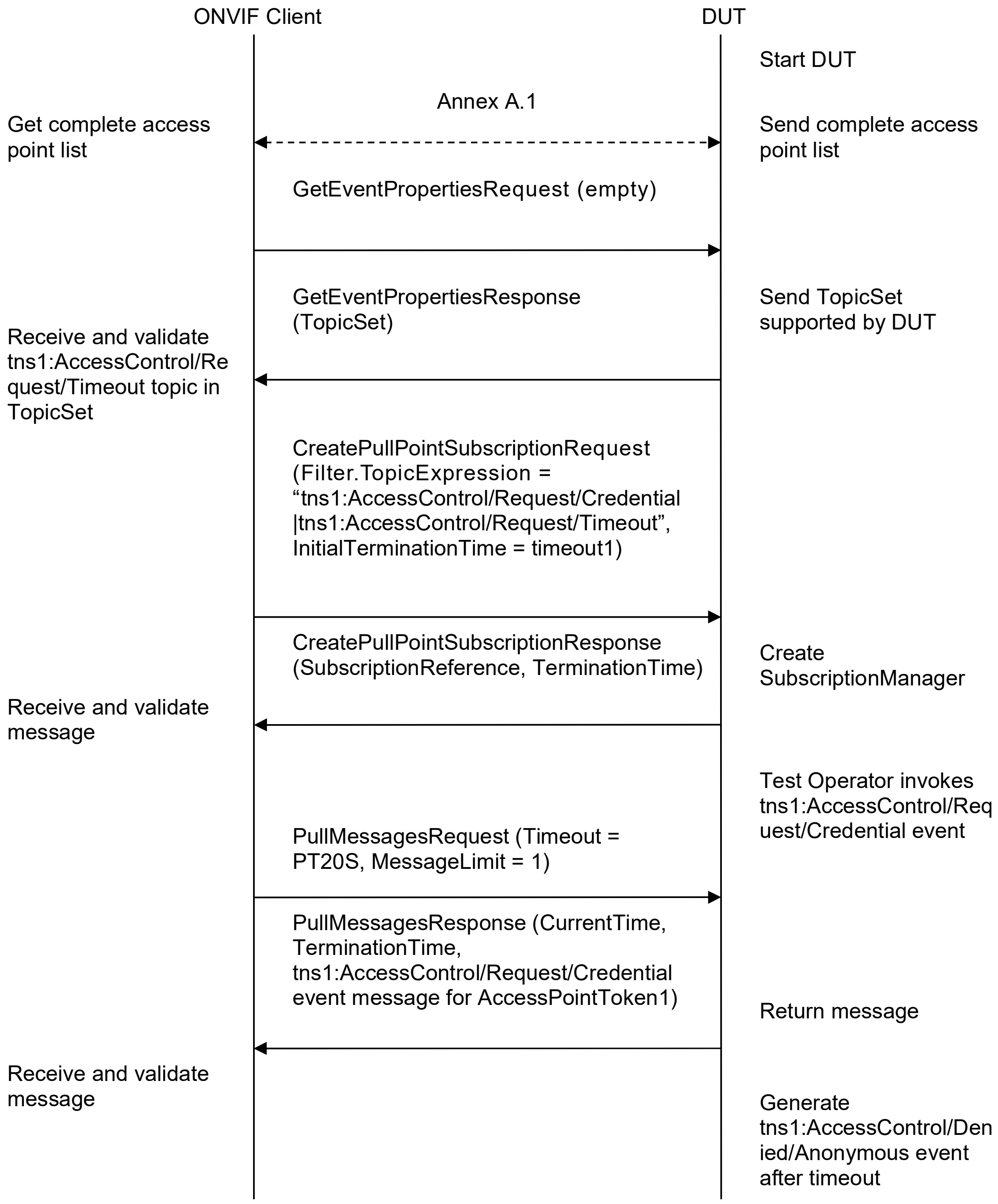
WSDL Reference: event.wsdl, accesscontrol.wsdl

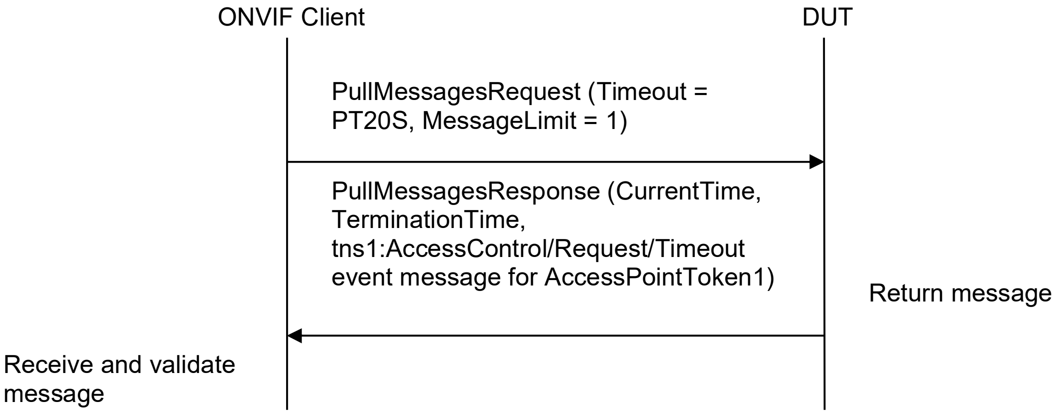
Test Purpose: To verify Access Timeout with Credential scenario in the case of External Authorization.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with External Authorization capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/Request/Timeout topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/Request/Credential |tns1:AccessControl/Request/Timeout”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/Request/Credential event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Request/Credential event message for AccessPointToken1)
Return message
Receive and validate message
Generate tns1:AccessControl/Denied/Anonymous event after timeout


ONVIF Client
DUT
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Request/Timeout event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. Get complete list of access points from the DUT (see Annex A.1).

4. If there are no Access Point with Capabilities.ExternalAuthorization = "true" in complete list of access points, fail the test and skip other steps.

5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

6. Verify the GetEventPropertiesResponse message from the DUT.

7. Check if there is an event with Topic tns1:AccessControl/Request/Timeout. If there is no event with such Topic skip other steps, fail the test and go to the next test.

8. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

10. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/Request/Credential Topic and tns1:AccessControl/Request/Timeout Topic as Filter and an InitialTerminationTime of timeout1.

11. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

12. Test Operator will invoke tns1:AccessControl/Request/Credential event for any Access Points with Capabilities.ExternalAuthorization = "true".

13. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

14. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 13.

15. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

16. Verify that TopicExpression is equal to tns1:AccessControl/Request/Credential for the Notification message.

17. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to one of existing Access Point Tokens Capabilities.ExternalAuthorization = "true" (e.g. complete list of access points contains Access Point with the same token and this Access Point has Capabilities.ExternalAuthorization = "true").

18. Verify that the notification contains Data.SimpleItem item with Name = "CredentialToken" and Value with type is equal to pt:ReferenceToken.

19. Verify that the notification which contains Data.SimpleItem item with Name = "CredentialHolderName" contains Value with type is equal to xs:string.

20. ONVIF Client waits for timeout to initiate tns1:AccessControl/Request/Timeout event.

21. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

22. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 21.

23. Verify that TopicExpression is equal to tns1:AccessControl/Request/Timeout for the Notification message.

24. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

25. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to Access Point Token from tns1:AccessControl/Request/Credential notification message.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/Request/Credential event at step 14.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/Request/Timeout event at step 22.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value, invalid CredentialToken, invalid CredentialHolderName).

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/Request/Credential in GetEventPropertiesResponse.

The DUT did not return valid Topic tns1:AccessControl/Request/Timeout in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message is received for step 14 or 22 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.4.7 EXTERNAL AUTHORIZATION WITH INVALID TOKEN

Test Label: External Authorization Verification with Invalid Token.

Test Case ID: ACCESSCONTROL-11-1-7

ONVIF Core Specification Coverage: ExternalAuthorization (ONVIF Access Control Service Specification)

Command under test: ExternalAuthorization

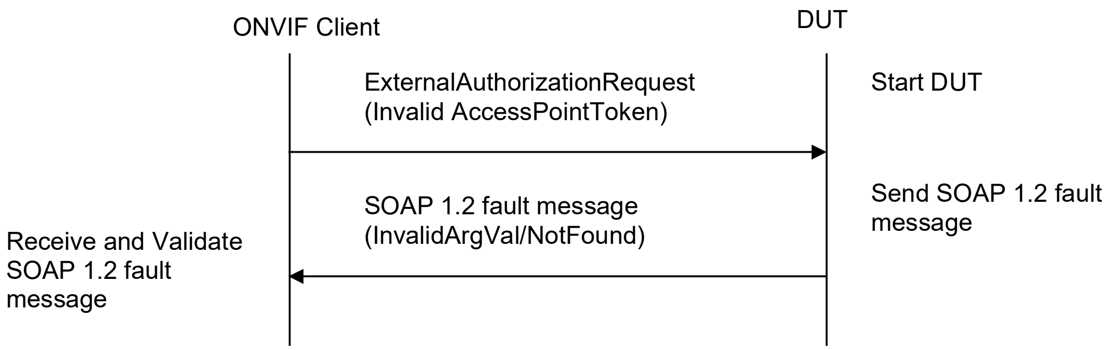
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify External Authorization with invalid Token.

Pre-Requisite: Access Control Service address was received from the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
ExternalAuthorizationRequest (Invalid AccessPointToken)
Start DUT
SOAP 1.2 fault message (InvalidArgVal/NotFound)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke ExternalAuthorizationRequest message **(invalid AccessPointToken**).
4. The DUT will generate SOAP 1.2 fault message (**InvalidArgVal/NotFound**).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT sent incorrect SOAP 1.2 fault message (fault code, namespace, etc.).

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

4.4.8 EXTERNAL AUTHORIZATION – COMMAND NOT SUPPORTED

Test Label: External Authorization Verification – Command not Supported.

Test Case ID: ACCESSCONTROL-11-1-8

ONVIF Core Specification Coverage: ExternalAuthorization (ONVIF Access Control Service Specification)

Command under test: ExternalAuthorization

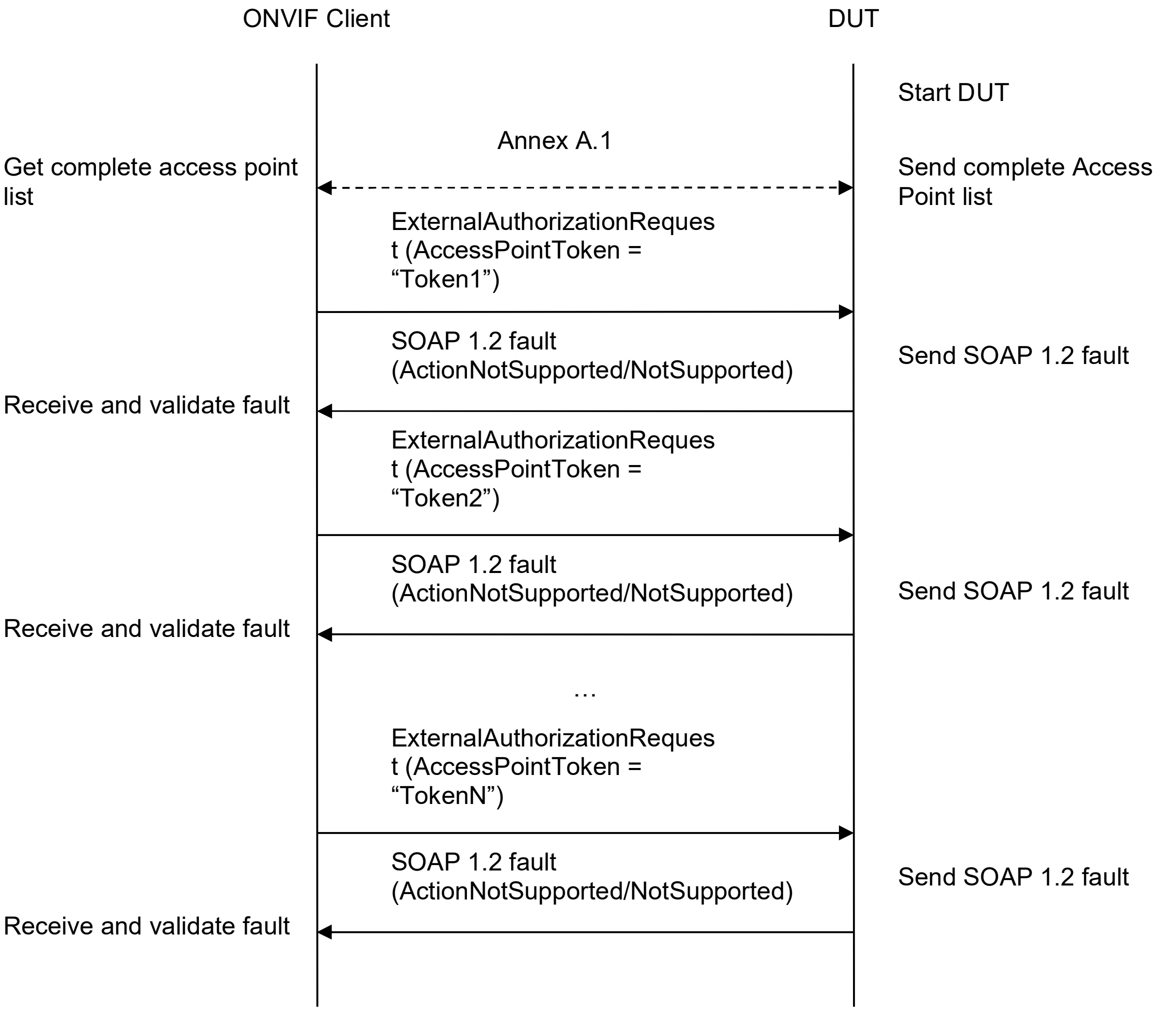
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify External Authorization in case Access Point does not support it.

Pre-Requisite: Access Control Service address was received from the DUT. At least one Access Point is configured without External Authorization capability and added to the DUT, if possible.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete Access Point list
ExternalAuthorizationRequest (AccessPointToken = “Token1”)
SOAP 1.2 fault (ActionNotSupported/NotSupported)
Send SOAP 1.2 fault
Receive and validate fault
ExternalAuthorizationRequest (AccessPointToken = “Token2”)
SOAP 1.2 fault (ActionNotSupported/NotSupported)
Send SOAP 1.2 fault
Receive and validate fault
…
ExternalAuthorizationRequest (AccessPointToken = “TokenN”)
SOAP 1.2 fault (ActionNotSupported/NotSupported)
Send SOAP 1.2 fault
Receive and validate fault

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. Get complete list of access points from the DUT (see Annex A.1).

4. If Access Point with **Token1** (Token1 is the first **AccessPointInfo.token** from the complete list of Access Points at step 3) has **AccessPointInfo.Capabilities.ExternalAuthorization** equal to **true**, then skip steps 5-6 and go to the step 7.

5. ONVIF Client will invoke ExternalAuthorizationRequest message (**AccessPointToken = “Token1”**, where Token1 is the first **AccessPointInfo.token** from the complete list of doors at step 3) to try perform external authorization.

6. The DUT will generate SOAP 1.2 fault message (**ActionNotSupported/NotSupported**).

7. Repeat steps 4-6 for all other tokens from the complete list of Access Point at step 3.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not return at least one Access Point on step 3.

Note: If the DUT does not return any Access Point for step 3, skip steps 4-7, fail the test and go to the next test.

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

4.5 Property Events

4.5.1 ACCESS CONTROL – ACCESS POINT ENABLED EVENT

Test Label: Access Control Service Property Events Check (Access Point Enabled).

Test Case ID: ACCESSCONTROL-5-1-1

ONVIF Core Specification Coverage: tns1:AccessPoint/State/Enabled (Access Control Service Specification), GetAccessPointState (ONVIF Access Control Service Specification), Properties (ONVIF Core Specification)

Command under test: GetAccessPointState

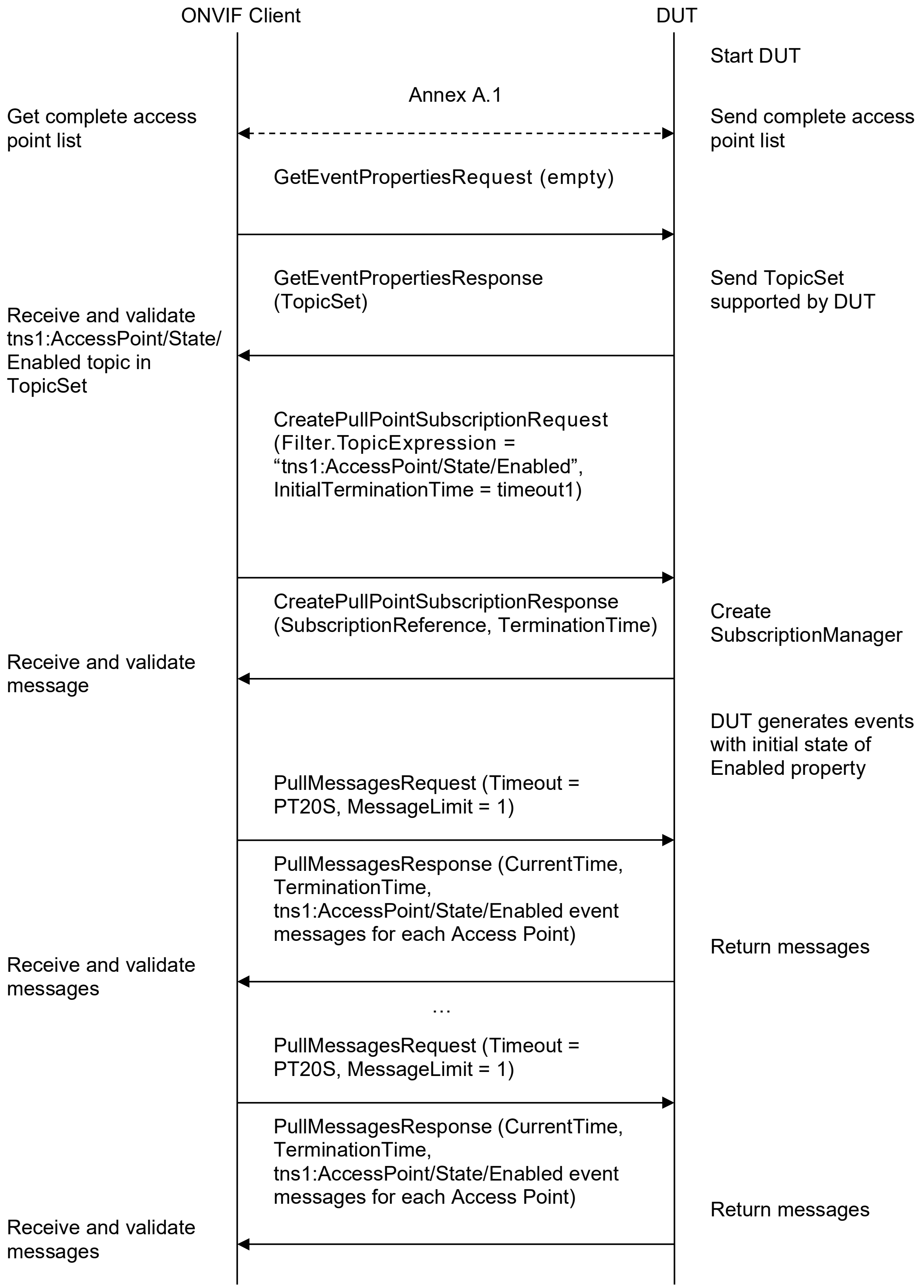
WSDL Reference: event.wsdl, accesscontrol.wsdl

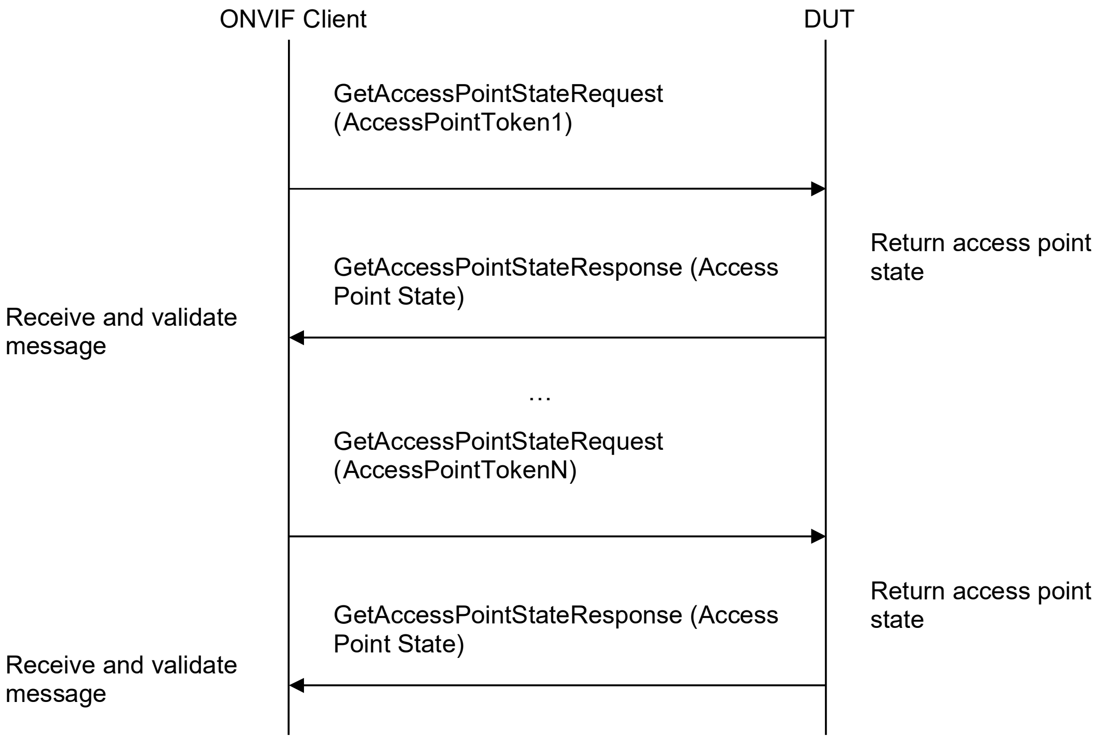
Test Purpose: To verify tns1:AccessPoint/State/Enabled event generation after subscription and to verify tns1:AccessPoint/State/Enabled event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with Enable/Disable capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessPoint/State/Enabled topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessPoint/State/Enabled”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
DUT generates events with initial state of Enabled property
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessPoint/State/Enabled event messages for each Access Point)
Return messages
Receive and validate messages
…
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessPoint/State/Enabled event messages for each Access Point)
Return messages
Receive and validate messages


ONVIF Client
DUT
GetAccessPointStateRequest (AccessPointToken1)
Return access point state
GetAccessPointStateResponse (Access Point State)
Receive and validate message
…
GetAccessPointStateRequest (AccessPointTokenN)
Return access point state
GetAccessPointStateResponse (Access Point State)
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. Get complete list of access points from the DUT (see Annex A.1).

4. If there is no Access Points with **DisableAccessPoint** equal to **true**, fail the test and skip other steps.

5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

6. Verify the GetEventPropertiesResponse message from the DUT.

7. Check if there is an event with Topic tns1:AccessPoint/State/Enabled. If there is no event with such Topic skip other steps, fail the test and go to the next test.

8. Check that this event is a Property event (MessageDescription.IsProperty = "true").

9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

10. Check that this event contains Data.SimpleItemDescription item with Name = "State" and Type = " xs:boolean".

11. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessPoint/State/Enabled Topic as Filter and an InitialTerminationTime of timeout1.

12. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

13. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

14. Verify that the DUT sends a PullMessagesResponse that contains NotificationMessages. Repeat step 13 until Notification for all Access Points will be received.

15. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).

16. Verify that TopicExpression is equal to tns1:AccessPoint/State/Enabled for all received Notification messages.

17. Verify that each notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to one of the existing Access Point Tokens (e.g. complete list of access points contains Access Point with the same token). Verify that there are Notification messages for each Access Point.

18. Verify that each notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to xs:boolean.

19. Verify that Notify PropertyOperation = "Initialized".

20. ONVIF Client will invoke GetAccessPointStateRequest message for each Access Point with corresponding tokens.

21. Verify the GetAccessPointStateResponse messages from the DUT. Verify that Data.SimpleItem item with Name = "Enabled" from Notification message has the same value with Enabled elements from corresponding GetAccessPointStateResponse messages for each AccessPoint.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send PullMessagesResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a Notification message that contains a property event tns1:AccessPoint/State/Enabled at least for one Access Point.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken or State values, PropertyOperation is not equal to "Initialized").

The DUT did not return at least one Access Point at step 3.

The DUT did not return Topic tns1:AccessPoint/State/Enabled in GetEventPropertiesResponse.

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

Note: ONVIF Client at step 14 will wait for Notification messages until notification for all Access Points is received or Operation Delay has expired. Notification messages for all Access Points are assumed as received, if the number of Notification messages is equal to the number of Access Points.

Note: If the DUT does not return any access point for step 3, skip steps 4-21, fail the test and go to the next test.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.5.2 ACCESS CONTROL – ACCESS POINT ENABLED EVENT STATE CHANGE

Test Label: Access Control Service Property Events State Change Check (Access Point Enabled).

Test Case ID: ACCESSCONTROL-5-1-2

ONVIF Core Specification Coverage: tns1:AccessPoint/State/Enabled (Access Control Service Specification), EnableAccessPoint (ONVIF Access Control Service Specification), DisableAccessPoint (ONVIF Access Control Service Specification), GetAccessPointState (ONVIF Access Control Service Specification)

Command under test: EnableAccessPoint, DisableAccessPoint, GetAccessPointState

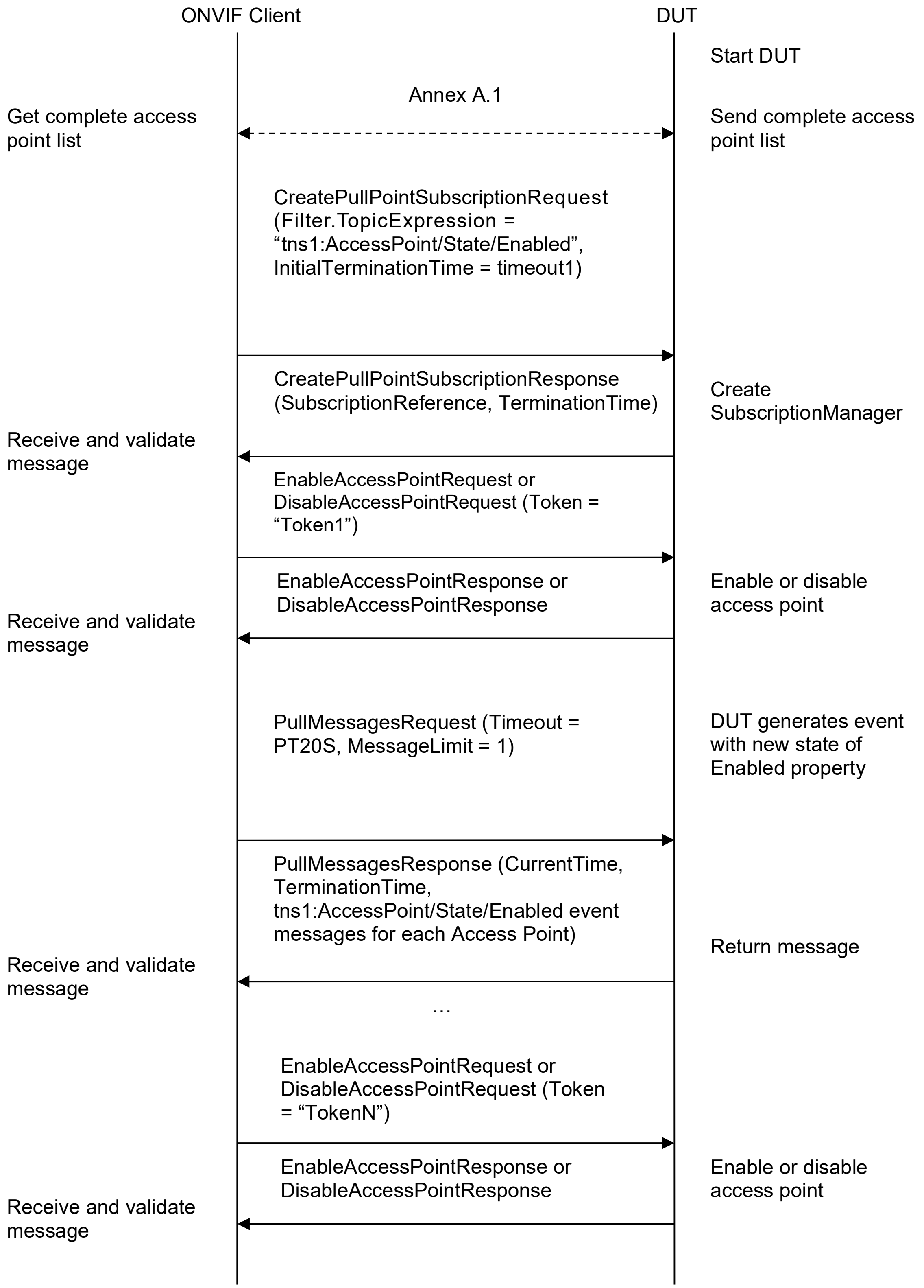
WSDL Reference: event.wsdl, accesscontrol.wsdl

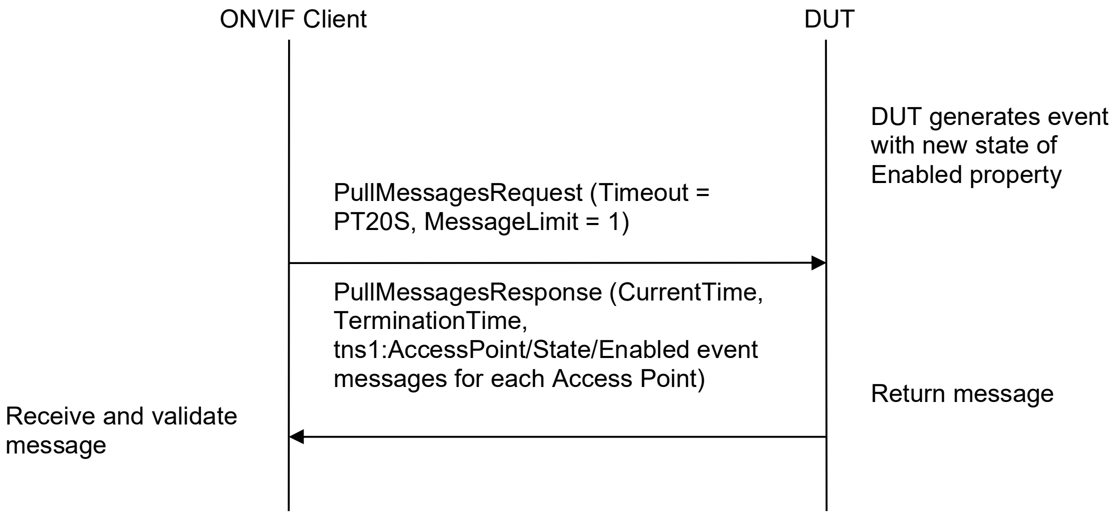
Test Purpose: To verify tns1:AccessPoint/State/Enabled event generation after property was changed and to verify tns1:AccessPoint/State/Enabled event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with Enable/Disable capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessPoint/State/Enabled”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
EnableAccessPointRequest or DisableAccessPointRequest (Token = “Token1”)
EnableAccessPointResponse or DisableAccessPointResponse
Enable or disable access point
Receive and validate message
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
DUT generates event with new state of Enabled property
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessPoint/State/Enabled event messages for each Access Point)
Return message
Receive and validate message
…
EnableAccessPointRequest or DisableAccessPointRequest (Token = “TokenN”)
EnableAccessPointResponse or DisableAccessPointResponse
Enable or disable access point
Receive and validate message


ONVIF Client
DUT
DUT generates event with new state of Enabled property
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessPoint/State/Enabled event messages for each Access Point)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. Get a complete list of access points from the DUT (see Annex A.1).

4. If there is no Access Points with **DisableAccessPoint** equal to **true**, fail the test and skip other steps.

5. If Access Point with **Token1** (Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) has **AccessPointInfo.Capabilities.DisableAccessPoint** equal to **false**, then skip steps 6-20 and go to the step 21.

6. ONVIF Client will invoke GetAccessPointStateRequest message (**TokenList.Token = Token1**, where Token1 is the first token from the complete list of access points at step 3) to retrieve Access Point state for specified token from the DUT.

7. Verify the GetAccessPointStateResponse message from the DUT.

8. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessPoint/State/Enabled Topic as Filter and an InitialTerminationTime of timeout1.

9. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

10. If Access Point with **Token1** (Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) has **AccessPointState.Enabled** equal to **true**, then skip steps 11-12 and go to the step 13.

11. ONVIF Client will invoke EnableAccessPointRequest message (**Token = "Token1"**, where Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) to try enabling access point.

12. Verify the EnableAccessPointResponse message from the DUT. Go to the step 15.

13. ONVIF Client will invoke DisableAccessPointRequest message (**Token = "Token1"**, where Token1 is the first **AccessPointInfo.token** from the complete list of access points at step 3) to try disabling access point.

14. Verify the DisableAccessPointResponse message from the DUT.

15. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

16. Verify that the DUT sends a PullMessagesResponse that contains NotificationMessages. Repeat step 15 until Notification with PropertyOperation = "Changed" for Access Point with Token = "Token1" will be received.

17. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).

18. Verify that TopicExpression is equal to tns1:AccessPoint/State/Enabled for all received Notification messages.

19. Verify that notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value = "**Token1"** (e.g. a complete list of access points contains Access Point with the same token).

20. Verify that notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to xs:boolean and with value equal to current state of Access Point.

21. Repeat steps 5-20 for all other tokens from complete list of access points at step 3.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send PullMessagesResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a Notification message that contains a property event tns1:AccessPoint/State/Enabled with valid AccessPointToken and Enabled value.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken or State values).

The DUT did not return at least one Access Point on step 3.

The DUT did not send valid EnableAccessPointResponse message.

The DUT did not send valid DisableAccessPointResponse message.

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

Note: All Notification messages except messages with PropertyOperation = "Changed" will be ignored.

Note: Test will be failed, if no required Notification messages are received for step 15 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

Note: If the DUT does not return any access point for step 3, skip steps 4-21, fail the test and go to the next test.

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

Note: The Renew has to be used for renew subscription during a test, if InitialTerminationTime expires.

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.6 Access granted events

4.6.1 ACCESS CONTROL – ACCESS GRANTED TO ANONYMOUS EVENT

Test Label: Access Control Service Events Check (AccessGranted/Anonymous).

Test Case ID: ACCESSCONTROL-6-1-1

ONVIF Core Specification Coverage: tns1:AccessControl/AccessGranted/Anonymous (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: none

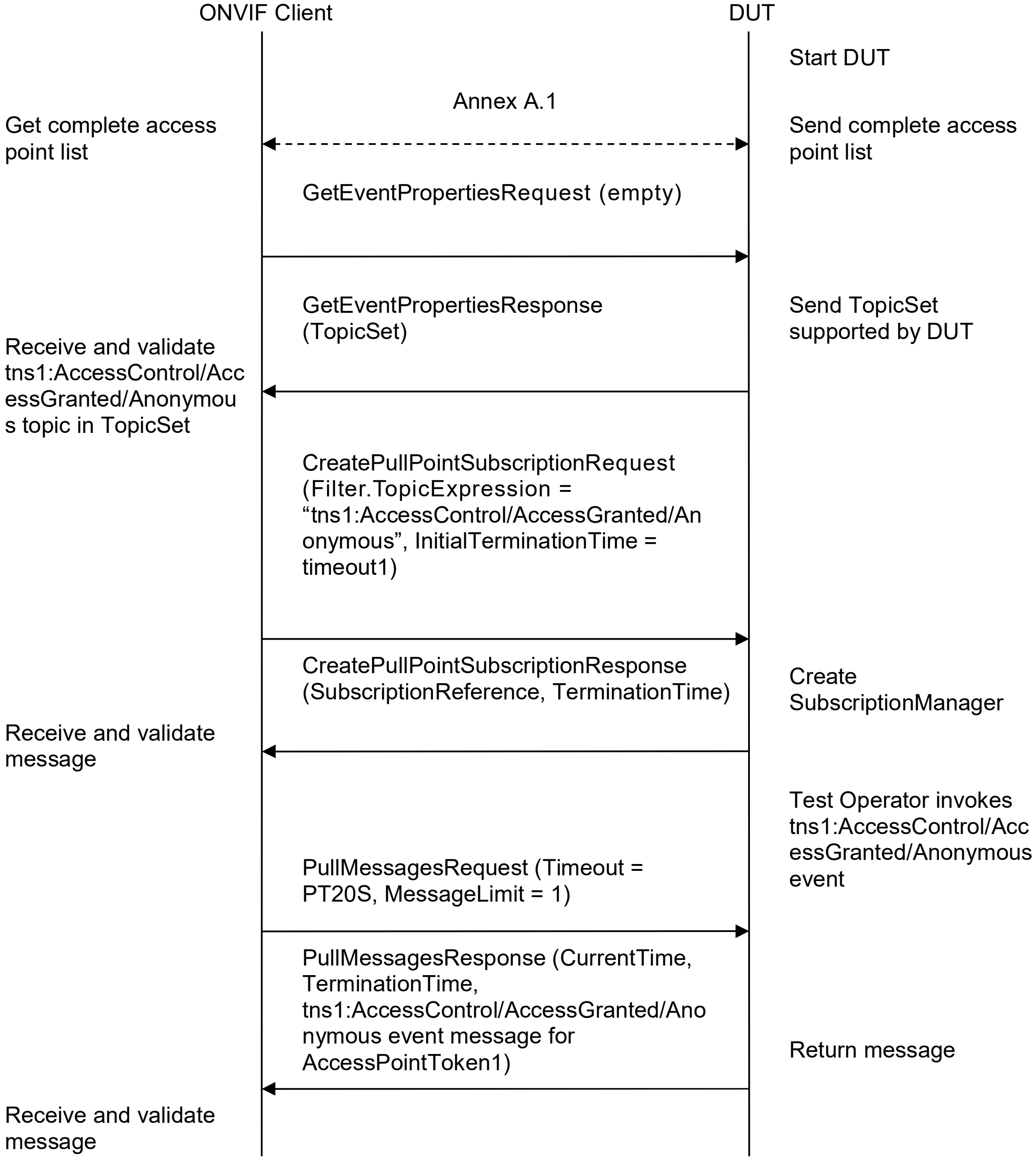
WSDL Reference: event.wsdl, accesscontrol.wsdl

Test Purpose: To verify tns1:AccessControl/AccessGranted/Anonymous event generation and to verify tns1:AccessControl/AccessGranted/Anonymous event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with Anonymous Access capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/AccessGranted/Anonymous topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/AccessGranted/Anonymous”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/AccessGranted/Anonymous event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/AccessGranted/Anonymous event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get complete list of access points from the DUT (see Annex A.1).

4. If there are no Access Point with Capabilities.AnonymousAccess = “true” in complete list of access points, fail the test and skip other steps.

5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

6. Verify the GetEventPropertiesResponse message from the DUT.

7. Check if there is an event with Topic tns1:AccessControl/AccessGranted/Anonymous. If there is no event with such Topic skip other steps, fail the test and go to the next test.

8. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

10. Check that if this event contains Data.SimpleItemDescription item with Name = "External", than it has Type = "xs:boolean".

11. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/AccessGranted/Anonymous Topic as Filter and an InitialTerminationTime of timeout1.

12. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

13. Test Operator will invoke tns1:AccessControl/AccessGranted/Anonymous event for any Access Points with Capabilities.AnonymousAccess = “true”.

14. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

15. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 14.

16. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

17. Verify that TopicExpression is equal to tns1:AccessControl/AccessGranted/Anonymous for the Notification message.

18. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to one of existing Access Point Tokens with Capabilities.AnonymousAccess = “true” (e.g. complete list of access points contains Access Point with the same token and this Access Point has Capabilities.AnonymousAccess = “true”).

19. Verify that the notification doesn’t contain Data.SimpleItem item with Name = "External", if it was not included in Topic in GetEventPropertiesResponse message. Verify that the notification doesn’t contain Data.SimpleItem item with Name = "External" or contains Data.SimpleItem item with Name = "External" and its Value is equal to “false”, if it was included in Topic in GetEventPropertiesResponse message.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/AccessGranted/Anonymous event.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value, invalid External).

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/AccessGranted/Anonymous in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message is received for step 15 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.6.2 ACCESS CONTROL – ACCESS GRANTED WITH CREDENTIAL EVENT

Test Label: Access Control Service Events Check (AccessGranted/Credential).

Test Case ID: ACCESSCONTROL-6-1-2

ONVIF Core Specification Coverage: tns1:AccessControl/AccessGranted/Credential (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: none

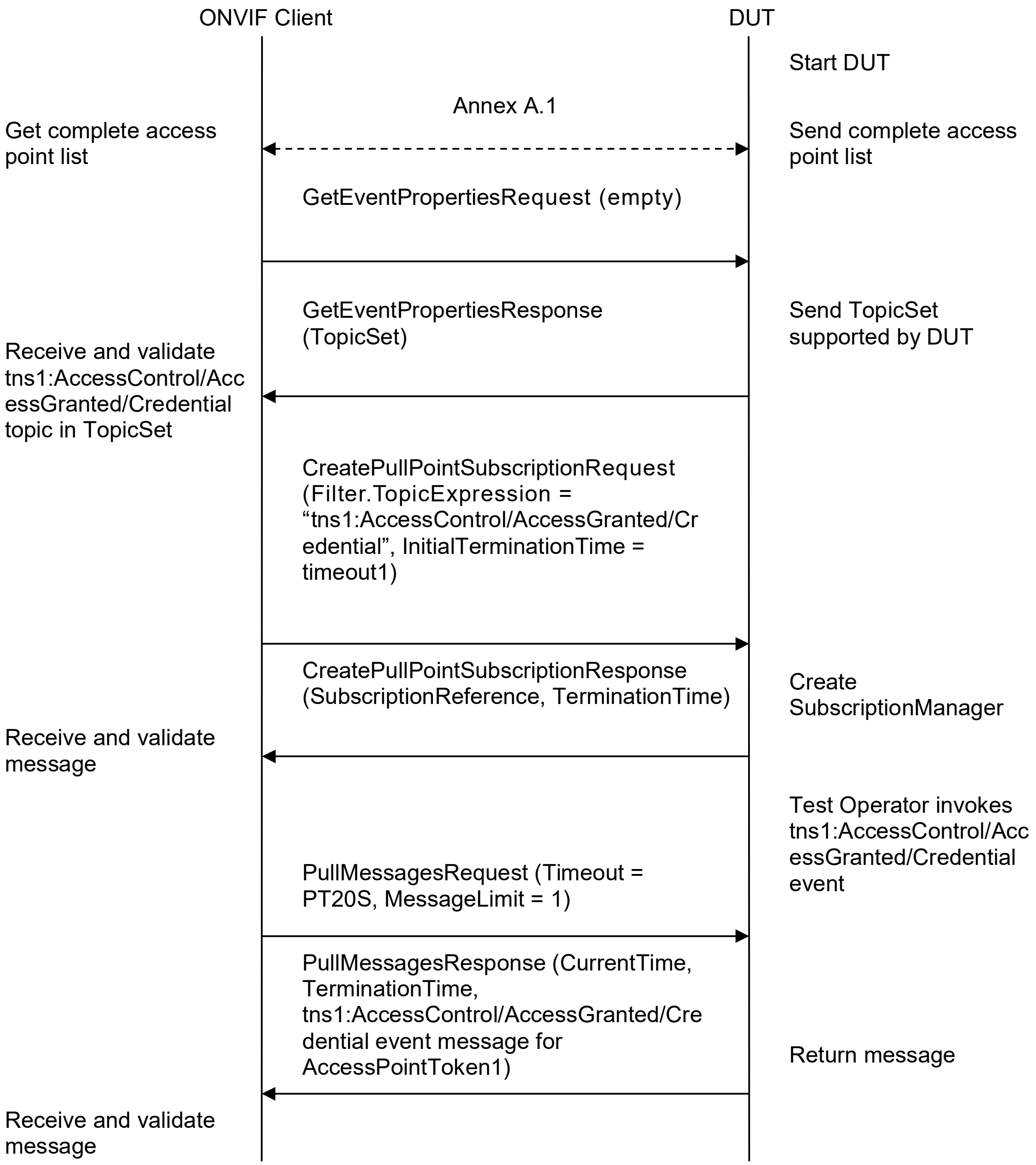
WSDL Reference: event.wsdl, accesscontrol.wsdl

Test Purpose: To verify tns1:AccessControl/AccessGranted/Credential event generation and to verify tns1:AccessControl/AccessGranted/Credential event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/AccessGranted/Credential topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/AccessGranted/Credential”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/AccessGranted/Credential event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/AccessGranted/Credential event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get complete list of access points from the DUT (see Annex A.1).

4. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

5. Verify the GetEventPropertiesResponse message from the DUT.

6. Check if there is an event with Topic tns1:AccessControl/AccessGranted/Credential. If there is no event with such Topic skip other steps, fail the test and go to the next test.

7. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

8. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

9. Check that this event contains Data.SimpleItemDescription item with Name = "CredentialToken" and Type = "pt:ReferenceToken".

10. Check that if this event contains Data.SimpleItemDescription item with Name = "CredentialHolderName", than it has Type = " xs:string".

11. Check that if this event contains Data.SimpleItemDescription item with Name = "External", than it has Type = "xs:boolean".

12. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/AccessGranted/Credential Topic as Filter and an InitialTerminationTime of timeout1.

13. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

14. Test Operator will invoke tns1:AccessControl/AccessGranted/Credential event for any Access Points.

15. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

16. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 15.

17. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

18. Verify that TopicExpression is equal to tns1:AccessControl/AccessGranted/Credential for the Notification message.

19. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to one of existing Access Point Tokens (e.g. complete list of access points contains Access Point with the same token).

20. Verify that the notification contains Data.SimpleItem item with Name = "CredentialToken" and Value with type is equal to pt:ReferenceToken.

21. Verify that the notification which contains Data.SimpleItem item with Name = "CredentialHolderName" contains Value with type is equal to xs:string.

22. Verify that the notification doesn’t contain Data.SimpleItem item with Name = "External", if it was not included in Topic in GetEventPropertiesResponse message. Verify that the notification doesn’t contain Data.SimpleItem item with Name = "External" or contains Data.SimpleItem item with Name = "External" and its Value is equal to “false”, if it was included in Topic in GetEventPropertiesResponse message.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/AccessGranted/Credential event.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value, invalid CredentialToken, invalid CredentialHolderName, invalid External).

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/AccessGranted/Credential in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message is received for step 16 during certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.7 Access taken events

4.7.1 ACCESS CONTROL – ACCESS TAKEN BY ANONYMOUS EVENT

Test Label: Access Control Service Events Check (AccessTaken/Anonymous).

Test Case ID: ACCESSCONTROL-7-1-1

ONVIF Core Specification Coverage: tns:AccessControl/AccessGranted/Anonymous (Access Control Service Specification), AccessControl/AccessTaken/Anonymous (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: none

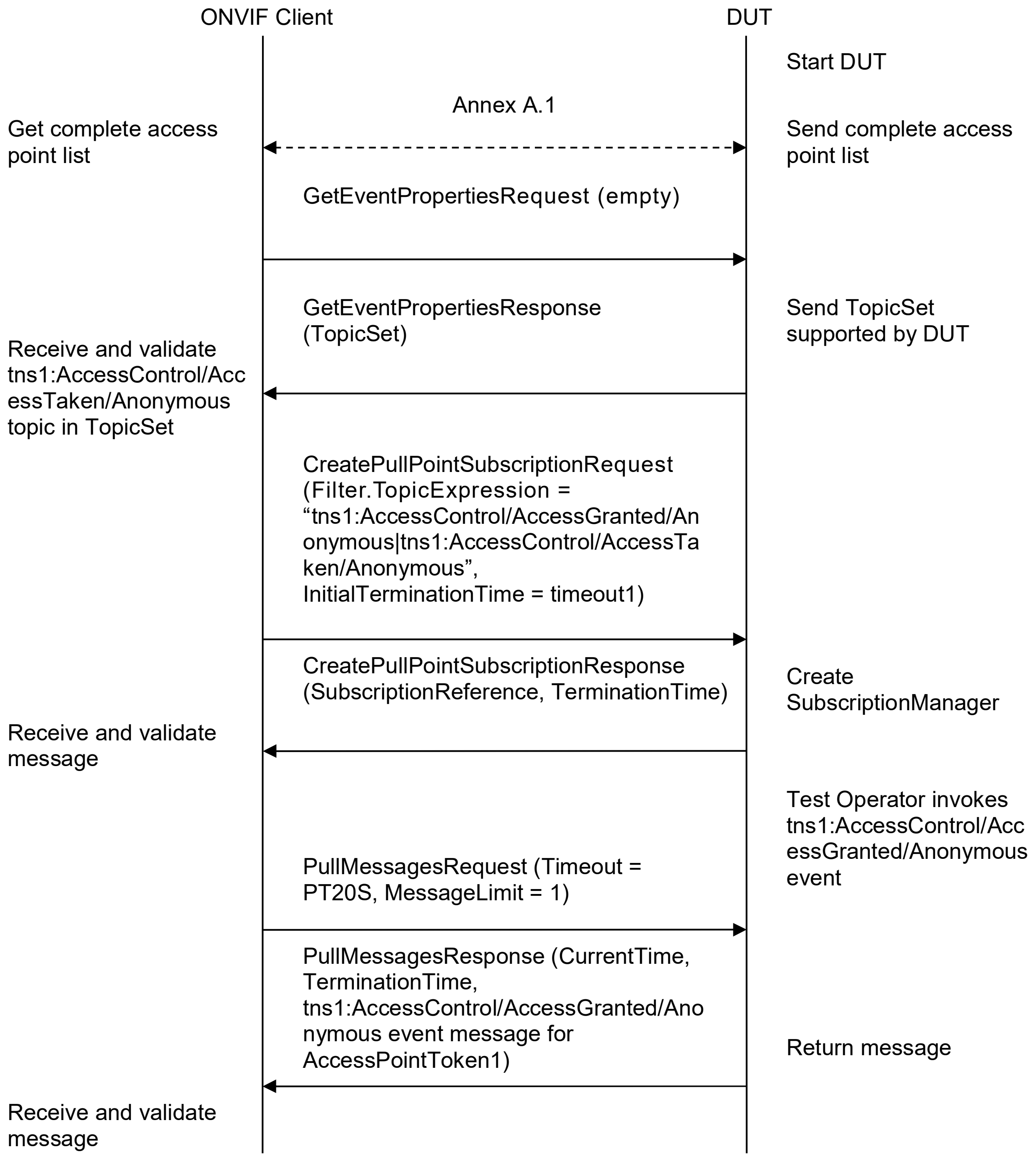
WSDL Reference: event.wsdl, accesscontrol.wsdl

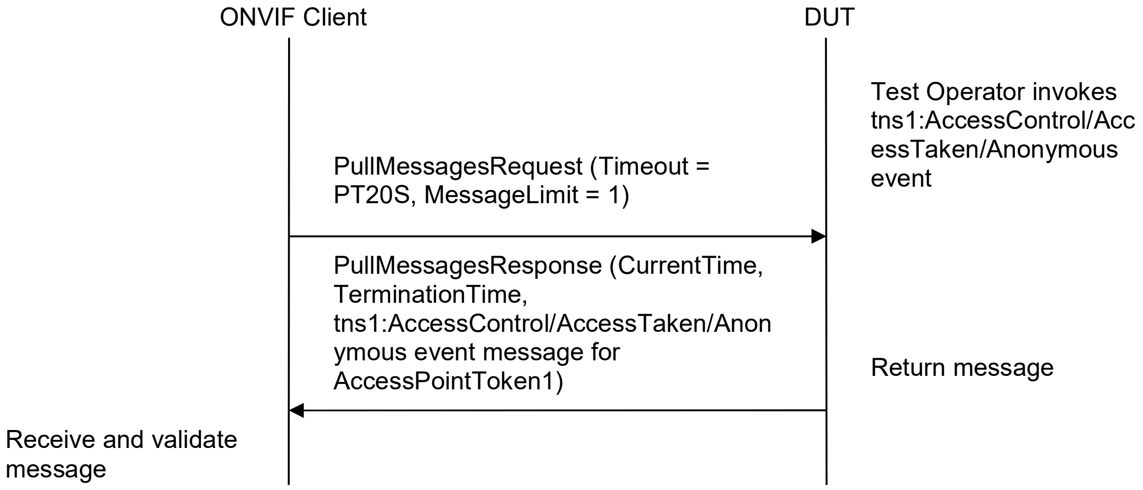
Test Purpose: To verify tns1:AccessControl/AccessTaken/Anonymous event generation and to verify tns1:AccessControl/AccessTaken/Anonymous event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with Anonymous Access capability and Access Taken capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/AccessTaken/Anonymous topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/AccessGranted/Anonymous|tns1:AccessControl/AccessTaken/Anonymous”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/AccessGranted/Anonymous event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/AccessGranted/Anonymous event message for AccessPointToken1)
Return message
Receive and validate message


ONVIF Client
DUT
Test Operator invokes tns1:AccessControl/AccessTaken/Anonymous event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/AccessTaken/Anonymous event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get complete list of access points from the DUT (see Annex A.1).
4. If there are no Access Point with Capabilities.AnonymousAccess = "true" and Capabilities.AccessTaken = "true" in complete list of access points, fail the test and skip other steps.
5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.
6. Verify the GetEventPropertiesResponse message from the DUT.
7. Check if there is an event with Topic tns1:AccessControl/AccessTaken/Anonymous. If there is no event with such Topic skip other steps, fail the test and go to the next test.
8. Check that this event is not a Property event (MessageDescription.IsProperty = "false").
9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".
10. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/AccessGranted/Anonymous Topic and tns1:AccessControl/AccessTaken/Anonymous Topic as Filter and an InitialTerminationTime of timeout1.
11. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
12. Test Operator will invoke tns1:AccessControl/AccessGranted/Anonymous event for any Access Points with Capabilities.AnonymousAccess = “true” and Capabilities.AccessTaken = "true".
13. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

14. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 13.

15. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

16. Verify that TopicExpression is equal to tns1:AccessControl/AccessGranted/Anonymous for the Notification message.

17. Test Operator will invoke tns1:AccessControl/AccessTaken/Anonymous event for the same Access Point.

18. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

19. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 18.

20. Verify that TopicExpression is equal to tns1:AccessControl/AccessTaken/Anonymous for the Notification message.

21. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to AccessPointToken from tns1:AccessControl/AccessGranted/Anonymous notification and there is Access Point Tokens with Capabilities.AnonymousAccess = "true" and Capabilities.AccessTaken = "true" in the complete list of access points (e.g. complete list of access points contains Access Point with the same token and this Access Point has Capabilities.AnonymousAccess = "true" and Capabilities.AccessTaken = "true").

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/AccessGranted/Anonymous event on step 14.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/AccessTaken/Anonymous event on step 19.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value) on step 19.

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/AccessTaken/Anonymous in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message is received for steps 14 and 19 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.7.2 ACCESS CONTROL – ACCESS TAKEN WITH CREDENTIAL EVENT

Test Label: Access Control Service Events Check (AccessTaken/Credential).

Test Case ID: ACCESSCONTROL-7-1-2

ONVIF Core Specification Coverage: tns:AccessControl/AccessGranted/Credential (Access Control Service Specification), tns1:AccessControl/AccessTaken/Credential (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: none

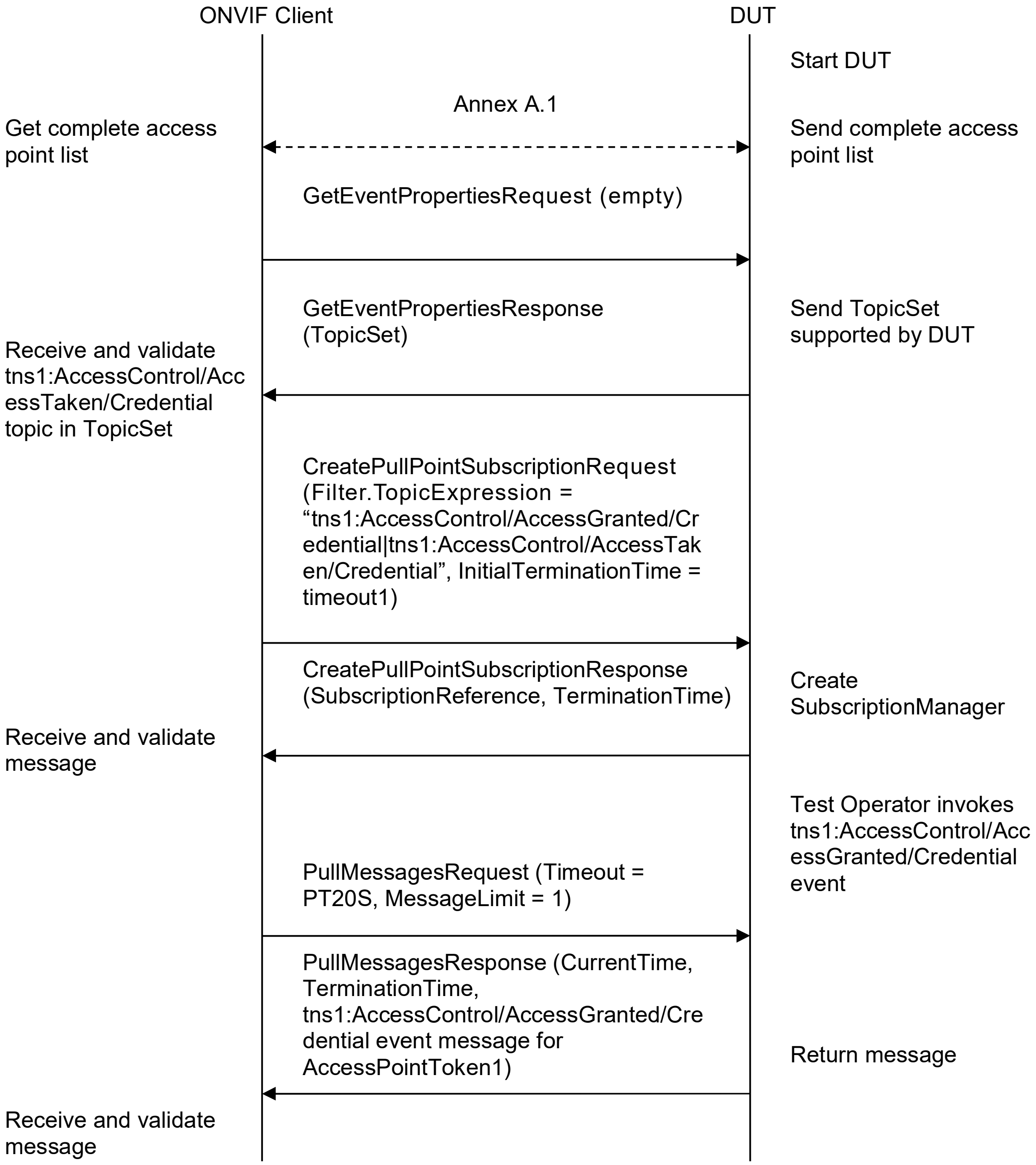
WSDL Reference: event.wsdl, accesscontrol.wsdl

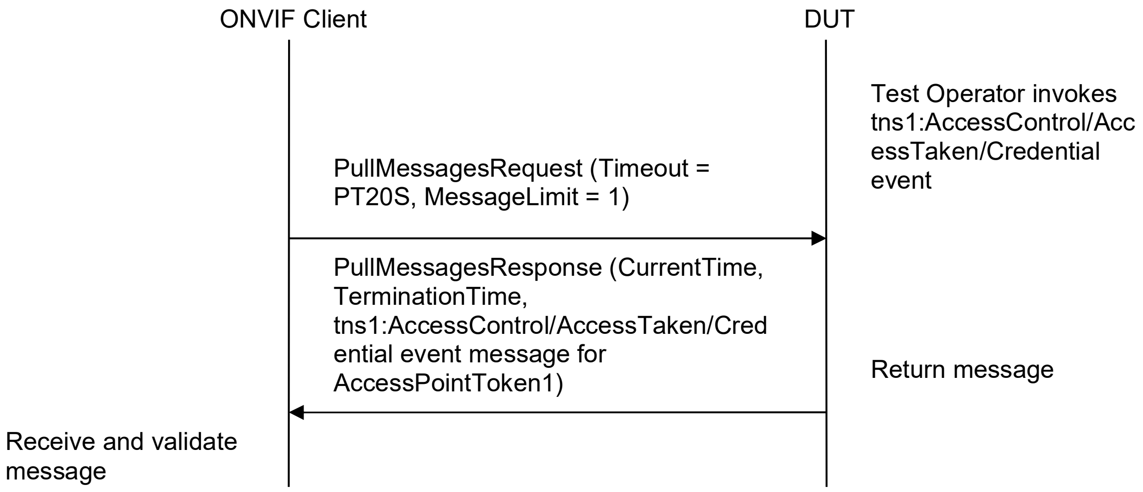
Test Purpose: To verify tns1:AccessControl/AccessTaken/Credential event generation and to verify tns1:AccessControl/AccessTaken/Credential event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with Access Taken capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/AccessTaken/Credential topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/AccessGranted/Credential|tns1:AccessControl/AccessTaken/Credential”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/AccessGranted/Credential event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/AccessGranted/Credential event message for AccessPointToken1)
Return message
Receive and validate message


ONVIF Client
DUT
Test Operator invokes tns1:AccessControl/AccessTaken/Credential event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/AccessTaken/Credential event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. Get complete list of access points from the DUT (see Annex A.1).

4. If there are no Access Points with Capabilities.AccessTaken = "true" in complete list of access points, fail the test and skip other steps.

5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

6. Verify the GetEventPropertiesResponse message from the DUT.

7. Check if there is an event with Topic tns1:AccessControl/AccessTaken/Credential. If there is no event with such Topic skip other steps, fail the test and go to the next test.

8. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

10. Check that this event contains Data.SimpleItemDescription item with Name = "CredentialToken" and Type = "pt:ReferenceToken".

11. Check that if this event contains Data.SimpleItemDescription item with Name = "CredentialHolderName", than it has Type = " xs:string".

12. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/AccessGranted/Credential Topic and tns1:AccessControl/AccessTaken/Credential Topic as Filter and an InitialTerminationTime of timeout1.

13. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

14. Test Operator will invoke tns1:AccessControl/AccessGranted/Credential event for any Access Points with Capabilities.AccessTaken = "true".

15. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

16. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 15.

17. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

18. Verify that TopicExpression is equal to tns1:AccessControl/AccessGranted/Credential for the Notification message.

19. Test Operator will invoke tns1:AccessControl/AccessTaken/Credential event for the same Access Point.

20. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

21. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 20.

22. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

23. Verify that TopicExpression is equal to tns1:AccessControl/AccessTaken/Credential for the Notification message.

24. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to AccessPointToken from tns1:AccessControl/AccessGranted/Credential notification and there is Access Point Tokens with Capabilities.AccessTaken = "true" in the complete list of access points (e.g. complete list of access points contains Access Point with the same token and this Access Point has Capabilities.AccessTaken = "true").

25. Verify that the notification contains Data.SimpleItem item with Name = "CredentialToken" and Value with type is equal to pt:ReferenceToken. Verify that this value is equal to the same Data.SimpleItem from tns1:AccessControl/AccessGranted/Credential notification.

26. Verify that the notification which contains Data.SimpleItem item with Name = "CredentialHolderName" contains Value with type is equal to xs:string. Verify that this value is equal to the same Data.SimpleItem from tns1:AccessControl/AccessGranted/Credential notification.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/AccessGranted/Credential event on step 16.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/AccessTaken/Credential event on step 21.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value, invalid CredentialToken value, and invalid CredentialHolderName value) on step 21.

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/AccessTaken/Credential in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message is received for step 16 and 19 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.8 Access not taken events

4.8.1 ACCESS CONTROL – ACCESS NOT TAKEN BY ANONYMOUS EVENT

Test Label: Access Control Service Events Check (AccessNotTaken/Anonymous).

Test Case ID: ACCESSCONTROL-8-1-1

ONVIF Core Specification Coverage: tns:AccessControl/AccessGranted/Anonymous (Access Control Service Specification), AccessControl/AccessNotTaken/Anonymous (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: none

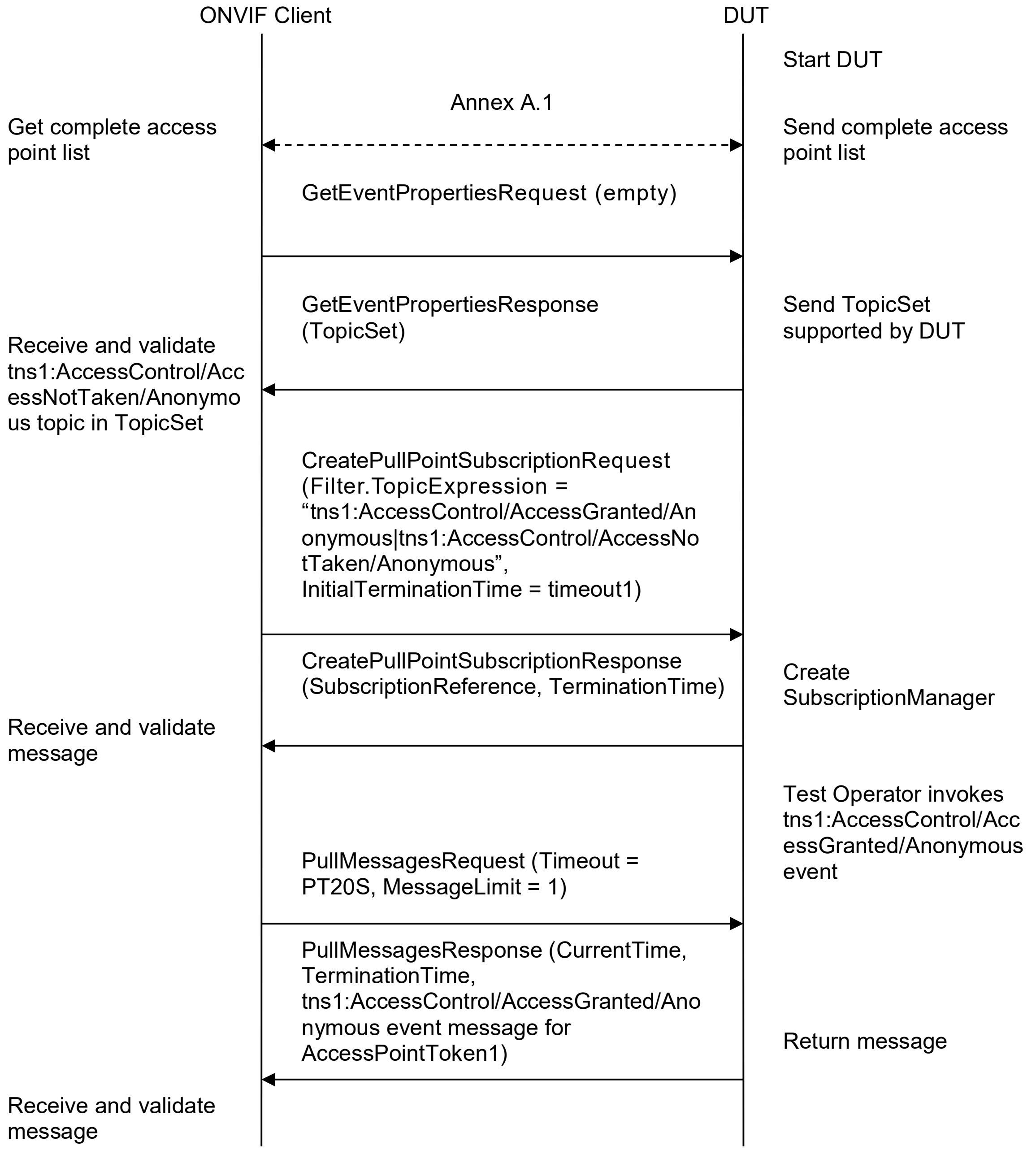
WSDL Reference: event.wsdl, accesscontrol.wsdl

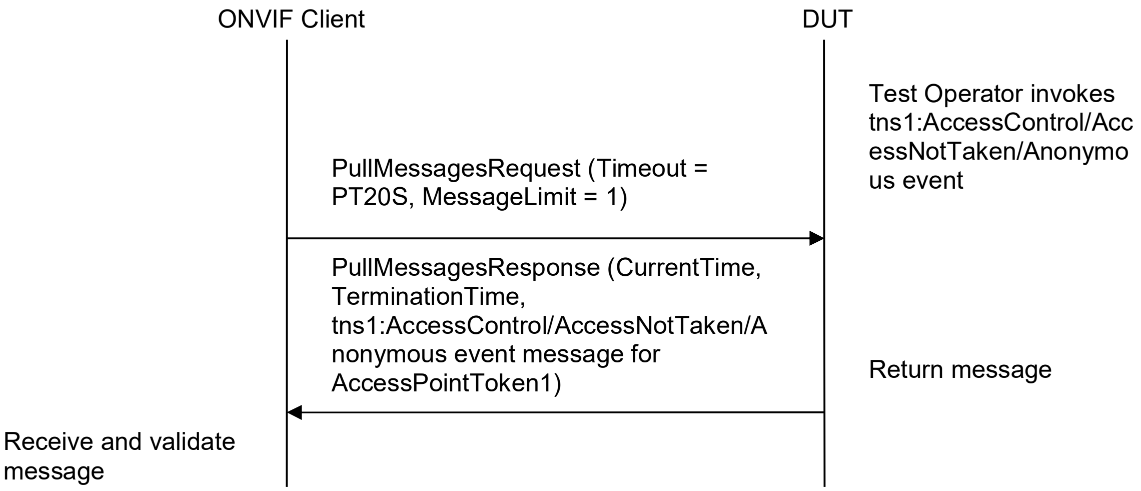
Test Purpose: To verify tns1:AccessControl/AccessNotTaken/Anonymous event generation and to verify tns1:AccessControl/AccessNotTaken/Anonymous event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with Anonymous Access capability and Access Taken capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/AccessNotTaken/Anonymous topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/AccessGranted/Anonymous|tns1:AccessControl/AccessNotTaken/Anonymous”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/AccessGranted/Anonymous event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/AccessGranted/Anonymous event message for AccessPointToken1)
Return message
Receive and validate message


ONVIF Client
DUT
Test Operator invokes tns1:AccessControl/AccessNotTaken/Anonymous event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/AccessNotTaken/Anonymous event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get complete list of access points from the DUT (see Annex A.1).
4. If there are no Access Points with Capabilities.AnonymousAccess = "true" and Capabilities.AccessTaken = "true" in complete list of access points, fail the test and skip other steps.
5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.
6. Verify the GetEventPropertiesResponse message from the DUT.
7. Check if there is an event with Topic tns1:AccessControl/AccessNotTaken/Anonymous. If there is no event with such Topic skip other steps, fail the test and go to the next test.
8. Check that this event is not a Property event (MessageDescription.IsProperty = "false").
9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".
10. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/AccessGranted/Anonymous Topic and tns1:AccessControl/AccessNotTaken/Anonymous Topic as Filter and an InitialTerminationTime of timeout1.
11. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
12. Test Operator will invoke tns1:AccessControl/AccessGranted/Anonymous event for any Access Points with Capabilities.AnonymousAccess = “true” and Capabilities.AccessTaken = "true".
13. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

14. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 13.

15. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

16. Verify that TopicExpression is equal to tns1:AccessControl/AccessGranted/Anonymous for the Notification message.

17. Test Operator will invoke tns1:AccessControl/AccessNotTaken/Anonymous event for the same Access Point.

18. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

19. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 18.

20. Verify that TopicExpression is equal to tns1:AccessControl/AccessNotTaken/Anonymous for the Notification message.

21. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to AccessPointToken from tns1:AccessControl/AccessGranted/Anonymous notification and there is Access Point Tokens with Capabilities.AnonymousAccess = "true" and Capabilities.AccessTaken = "true" in the complete list of access points (e.g. complete list of access points contains Access Point with the same token and this Access Point has Capabilities.AnonymousAccess = "true" and Capabilities.AccessTaken = "true").

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/AccessGranted/Anonymous event on step 14.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/AccessNotTaken/Anonymous event on step 19.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value) on step 19.

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/AccessNotTaken/Anonymous in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message is received for steps 14 and 19 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.8.2 ACCESS CONTROL – ACCESS NOT TAKEN WITH CREDENTIAL EVENT

Test Label: Access Control Service Events Check (AccessTaken/Credential).

Test Case ID: ACCESSCONTROL-8-1-2

ONVIF Core Specification Coverage: tns:AccessControl/AccessGranted/Credential (Access Control Service Specification), tns1:AccessControl/AccessNotTaken/Credential (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: none

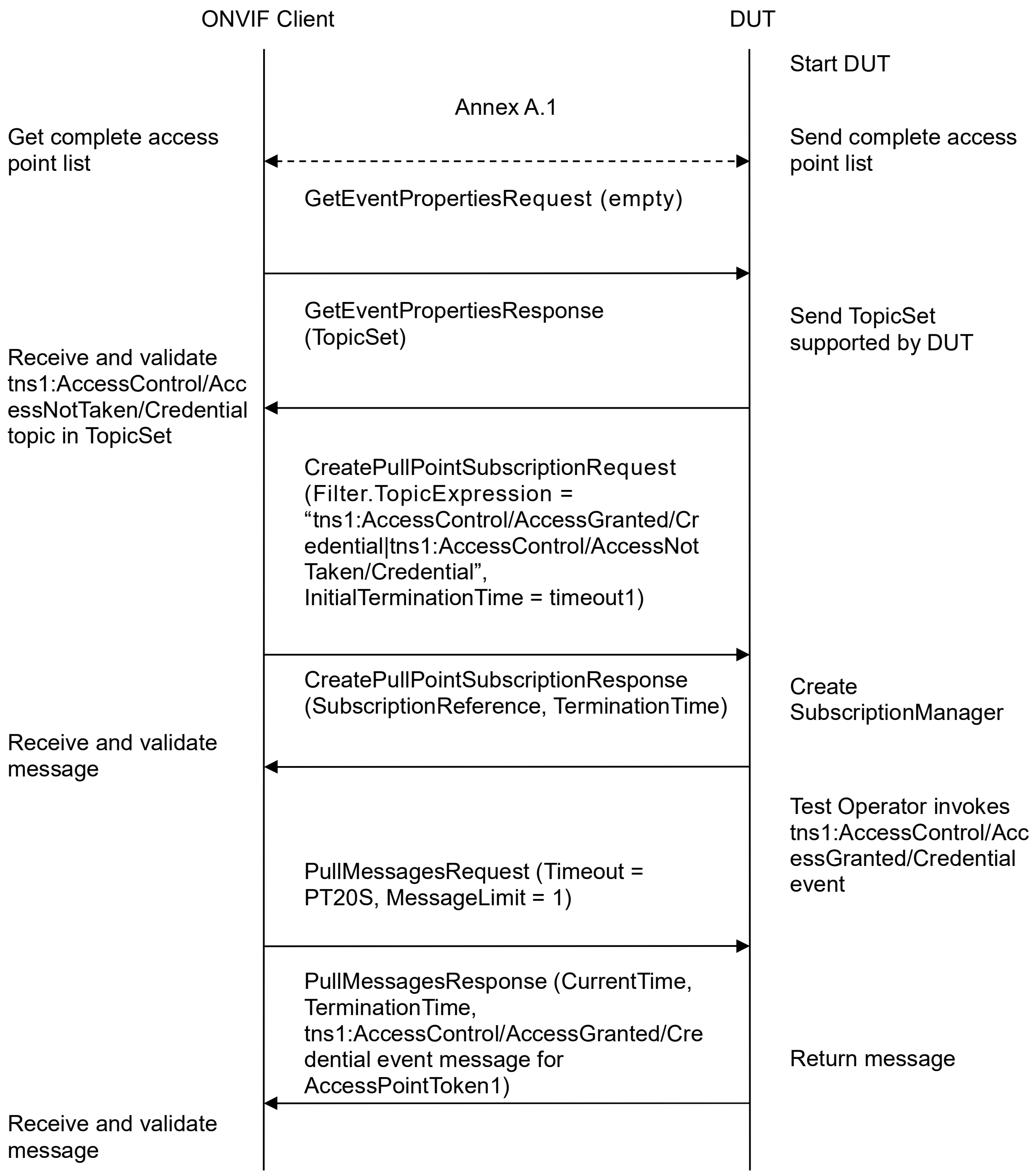
WSDL Reference: event.wsdl, accesscontrol.wsdl

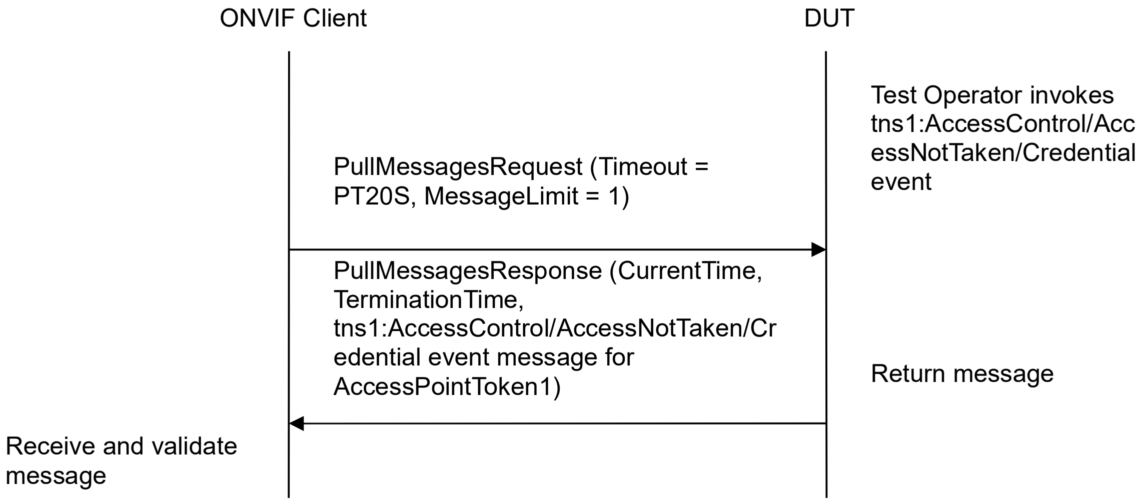
Test Purpose: To verify tns1:AccessControl/AccessNotTaken/Credential event generation and to verify tns1:AccessControl/AccessNotTaken/Credential event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with Access Taken capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/AccessNotTaken/Credential topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/AccessGranted/Credential|tns1:AccessControl/AccessNotTaken/Credential”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/AccessGranted/Credential event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/AccessGranted/Credential event message for AccessPointToken1)
Return message
Receive and validate message


ONVIF Client
DUT
Test Operator invokes tns1:AccessControl/AccessNotTaken/Credential event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/AccessNotTaken/Credential event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. Get complete list of access points from the DUT (see Annex A.1).

4. If there are no Access Point with Capabilities.AccessTaken = "true" in complete list of access points, fail the test and skip other steps.

5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

6. Verify the GetEventPropertiesResponse message from the DUT.

7. Check if there is an event with Topic tns1:AccessControl/AccessNotTaken/Credential. If there is no event with such Topic skip other steps, fail the test and go to the next test.

8. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

10. Check that this event contains Data.SimpleItemDescription item with Name = "CredentialToken" and Type = "pt:ReferenceToken".

11. Check that if this event contains Data.SimpleItemDescription item with Name = "CredentialHolderName", than it has Type = " xs:string".

12. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/AccessGranted/Credential Topic and tns1:AccessControl/AccessNotTaken/Credential Topic as Filter and an InitialTerminationTime of timeout1.

13. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

14. Test Operator will invoke tns1:AccessControl/AccessGranted/Credential event for any Access Points with Capabilities.AccessTaken = "true".

15. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

16. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 15.

17. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

18. Verify that TopicExpression is equal to tns1:AccessControl/AccessGranted/Credential for the Notification message.

19. Test Operator will invoke tns1:AccessControl/AccessNotTaken/Credential event for the same Access Point.

20. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

21. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 20.

22. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

23. Verify that TopicExpression is equal to tns1:AccessControl/AccessNotTaken/Credential for the Notification message.

24. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to AccessPointToken from tns1:AccessControl/AccessGranted/Credential notification and there is Access Point Tokens with Capabilities.AccessTaken = "true" in the complete list of access points (e.g. complete list of access points contains Access Point with the same token and this Access Point has Capabilities.AccessTaken = "true").

25. Verify that the notification contains Data.SimpleItem item with Name = "CredentialToken" and Value with type is equal to pt:ReferenceToken. Verify that this value is equal to the same Data.SimpleItem from tns1:AccessControl/AccessGranted/Credential notification.

26. Verify that the notification which contains Data.SimpleItem item with Name = "CredentialHolderName" contains Value with type is equal to xs:string. Verify that this value is equal to the same Data.SimpleItem from tns1:AccessControl/AccessGranted/Credential notification.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/AccessGranted/Credential event on step 16.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/AccessNotTaken/Credential event on step 21.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value, invalid CredentialToken value, and invalid CredentialHolderName value) on step 21.

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/AccessNotTaken/Credential in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime will expire.

Note: Test will be failed, if no Notification message is received for step 16 and 19 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.9 Access denied events

4.9.1 ACCESS CONTROL – ACCESS DENIED TO ANONYMOUS EVENT

Test Label: Access Control Service Events Check (Denied/Anonymous).

Test Case ID: ACCESSCONTROL-9-1-1

ONVIF Core Specification Coverage: tns1:AccessControl/Denied/Anonymous (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: none

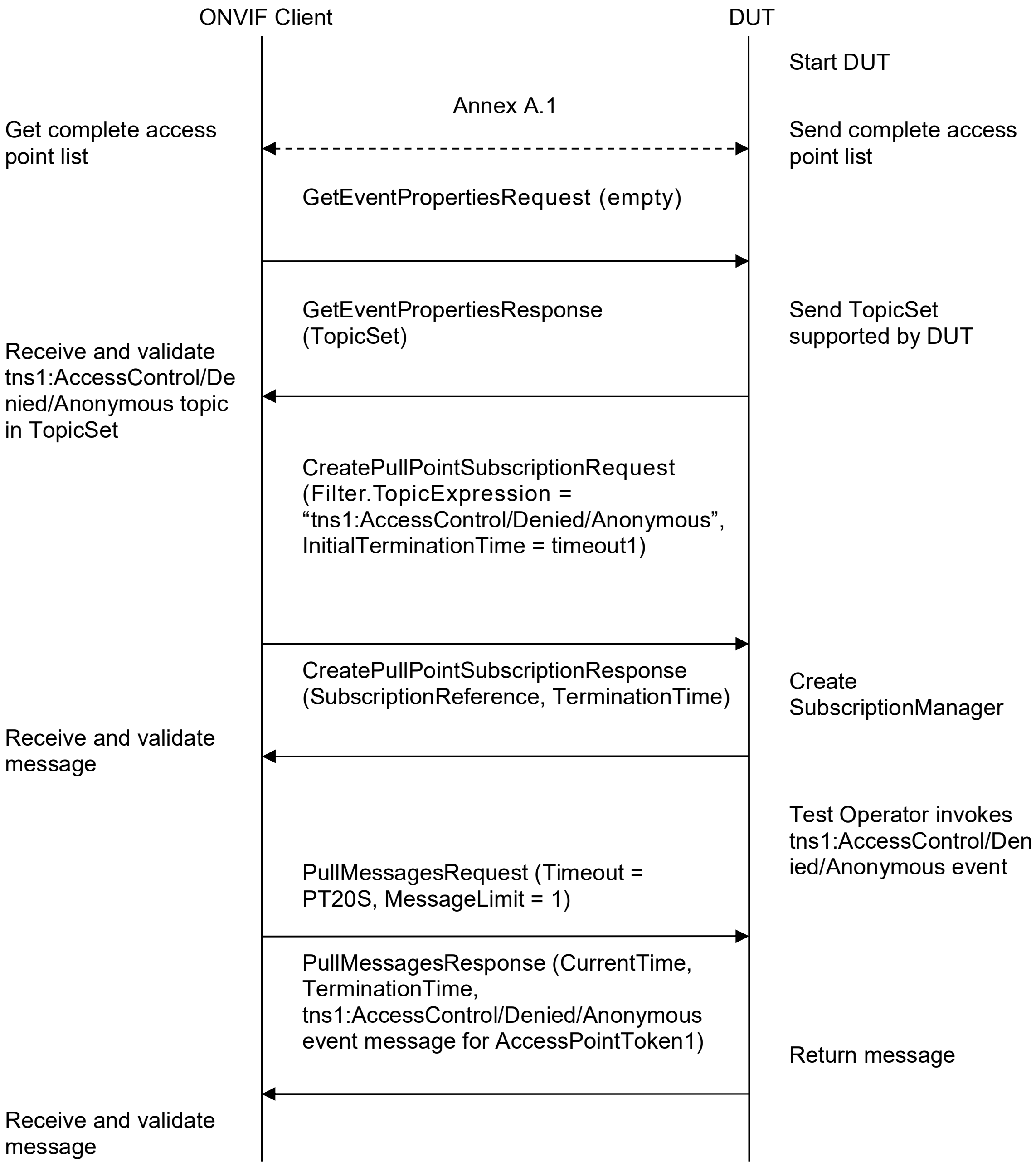
WSDL Reference: event.wsdl, accesscontrol.wsdl

Test Purpose: To verify tns1:AccessControl/Denied/Anonymous event generation and to verify tns1:AccessControl/Denied/Anonymous event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with Anonymous Access capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/Denied/Anonymous topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/Denied/Anonymous”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/Denied/Anonymous event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Denied/Anonymous event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get complete list of access points from the DUT (see Annex A.1).

4. If there are no Access Points with Capabilities.AnonymousAccess = “true” in complete list of access points, fail the test and skip other steps.

5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

6. Verify the GetEventPropertiesResponse message from the DUT.

7. Check if there is an event with Topic tns1:AccessControl/Denied/Anonymous. If there is no event with such Topic skip other steps, fail the test and go to the next test.

8. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

10. Check that if this event contains Data.SimpleItemDescription item with Name = "External", than it has Type = "xs:boolean".

11. Check that this event contains Data.SimpleItemDescription item with Name = "Reason" and Type = " xs:string".

12. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/Denied/Anonymous Topic as Filter and an InitialTerminationTime of timeout1.

13. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

14. Test Operator will invoke tns1:AccessControl/Denied/Anonymous event for any Access Points with Capabilities.AnonymousAccess = “true”.

15. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

16. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 15.

17. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

18. Verify that TopicExpression is equal to tns1:AccessControl/Denied/Anonymous for the Notification message.

19. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to one of existing Access Point Tokens with Capabilities.AnonymousAccess = “true” (e.g. complete list of access points contains Access Point with the same token and this Access Point has Capabilities.AnonymousAccess = “true”).

20. Verify that the notification doesn’t contain Data.SimpleItem item with Name = "External", if it was not included in Topic in GetEventPropertiesResponse message. Verify that the notification doesn’t contain Data.SimpleItem item with Name = "External" or contains Data.SimpleItem item with Name = "External" and its Value is equal to “false”, if it was included in Topic in GetEventPropertiesResponse message.

21. Verify that the notification contains Data.SimpleItem item with Name = "Reason" contains Value with type is equal to xs:string. Check that Reason value is one of the following strings: “CredentialNotEnabled”, “CredentialNotActive”, “CredentialExpired”, “InvalidPIN”, “NotPermittedAtThisTime”, “Unauthorized”, “Other”.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/Denied/Anonymous event.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value, invalid External, invalid Reason).

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/Denied/Anonymous in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message will be received for step 16 during certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.9.2 ACCESS CONTROL – ACCESS DENIED WITH CREDENTIAL EVENT

Test Label: Access Control Service Events Check (Denied/Credential).

Test Case ID: ACCESSCONTROL-9-1-2

ONVIF Core Specification Coverage: tns1:AccessControl/Denied/Credential (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: none

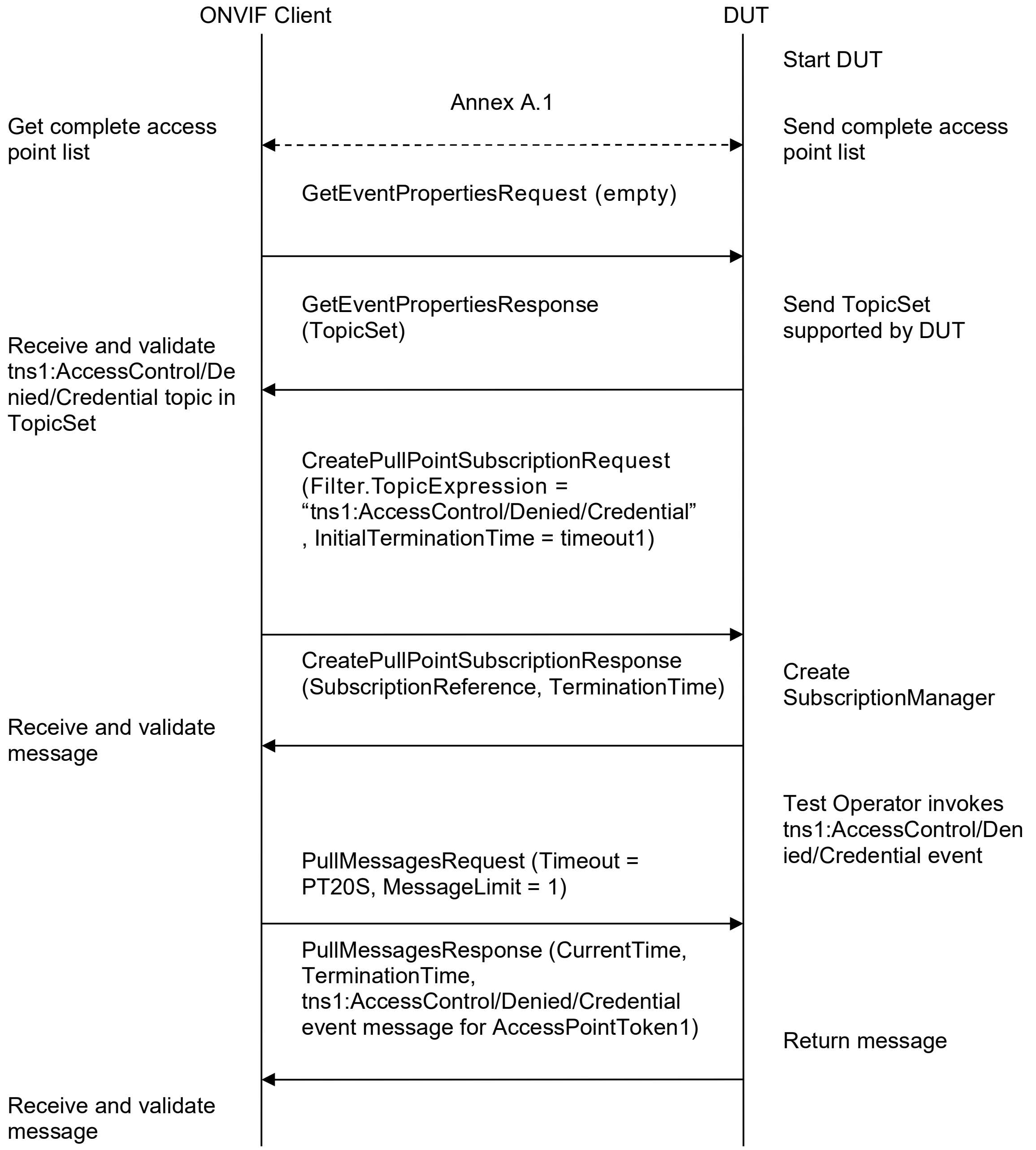
WSDL Reference: event.wsdl, accesscontrol.wsdl

Test Purpose: To verify tns1:AccessControl/Denied/Credential generation and to verify tns1:AccessControl/Denied/Credential event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/Denied/Credential topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/Denied/Credential” , InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/Denied/Credential event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Denied/Credential event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get complete list of access points from the DUT (see Annex A.1).
4. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.
5. Verify the GetEventPropertiesResponse message from the DUT.
6. Check if there is an event with Topic tns1:AccessControl/Denied/Credential. If there is no event with such Topic skip other steps, fail the test and go to the next test.
7. Check that this event is not a Property event (MessageDescription.IsProperty = "false").
8. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".
9. Check that this event contains Data.SimpleItemDescription item with Name = "CredentialToken" and Type = "pt:ReferenceToken".
10. Check that if this event contains Data.SimpleItemDescription item with Name = "CredentialHolderName", than it has Type = " xs:string".
11. Check that if this event contains Data.SimpleItemDescription item with Name = "External", than it has Type = "xs:boolean".
12. Check that this event contains Data.SimpleItemDescription item with Name = "Reason" and Type = " xs:string".
13. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/Denied/Credential Topic as Filter and an InitialTerminationTime of timeout1.
14. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
15. Test Operator will invoke tns1:AccessControl/Denied/Credential event for any Access Points.
16. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
17. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 16.
18. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).
19. Verify that TopicExpression is equal to tns1:AccessControl/Denied/Credential for the Notification message.
20. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to one of existing Access Point Tokens (e.g. complete list of access points contains Access Point with the same token).
21. Verify that the notification contains Data.SimpleItem item with Name = "CredentialToken" and Value with type is equal to pt:ReferenceToken.

22. Verify that the notification which contains Data.SimpleItem item with Name = "CredentialHolderName" contains Value with type is equal to xs:string.

23. Verify that the notification doesn’t contain Data.SimpleItem item with Name = "External", if it was not included in Topic in GetEventPropertiesResponse message. Verify that the notification doesn’t contain Data.SimpleItem item with Name = "External" or contains Data.SimpleItem item with Name = "External" and its Value is equal to “false”, if it was included in Topic in GetEventPropertiesResponse message.

24. Verify that the notification contains Data.SimpleItem item with Name = "Reason" contains Value with type is equal to xs:string. Check that Reason value is one of the following strings: “CredentialNotEnabled”, “CredentialNotActive”, “CredentialExpired”, “InvalidPIN”, “NotPermittedAtThisTime”, “Unauthorized”, “Other”.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/Denied/Credential event.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value, invalid CredentialToken, invalid CredentialHolderName, invalid External, invalid Reason).

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/Denied/Credential in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message is received for step 17 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.9.3 ACCESS CONTROL – ACCESS DENIED WITH CREDENTIAL EVENT (CREDENTIAL NOT FOUND – CARD)

Test Label: Access Control Service Events Check (Denied/CredentialNotFound/Card).

Test Case ID: ACCESSCONTROL-9-1-3

ONVIF Core Specification Coverage: tns1:AccessControl/Denied/CredentialNotFound/Card (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: none

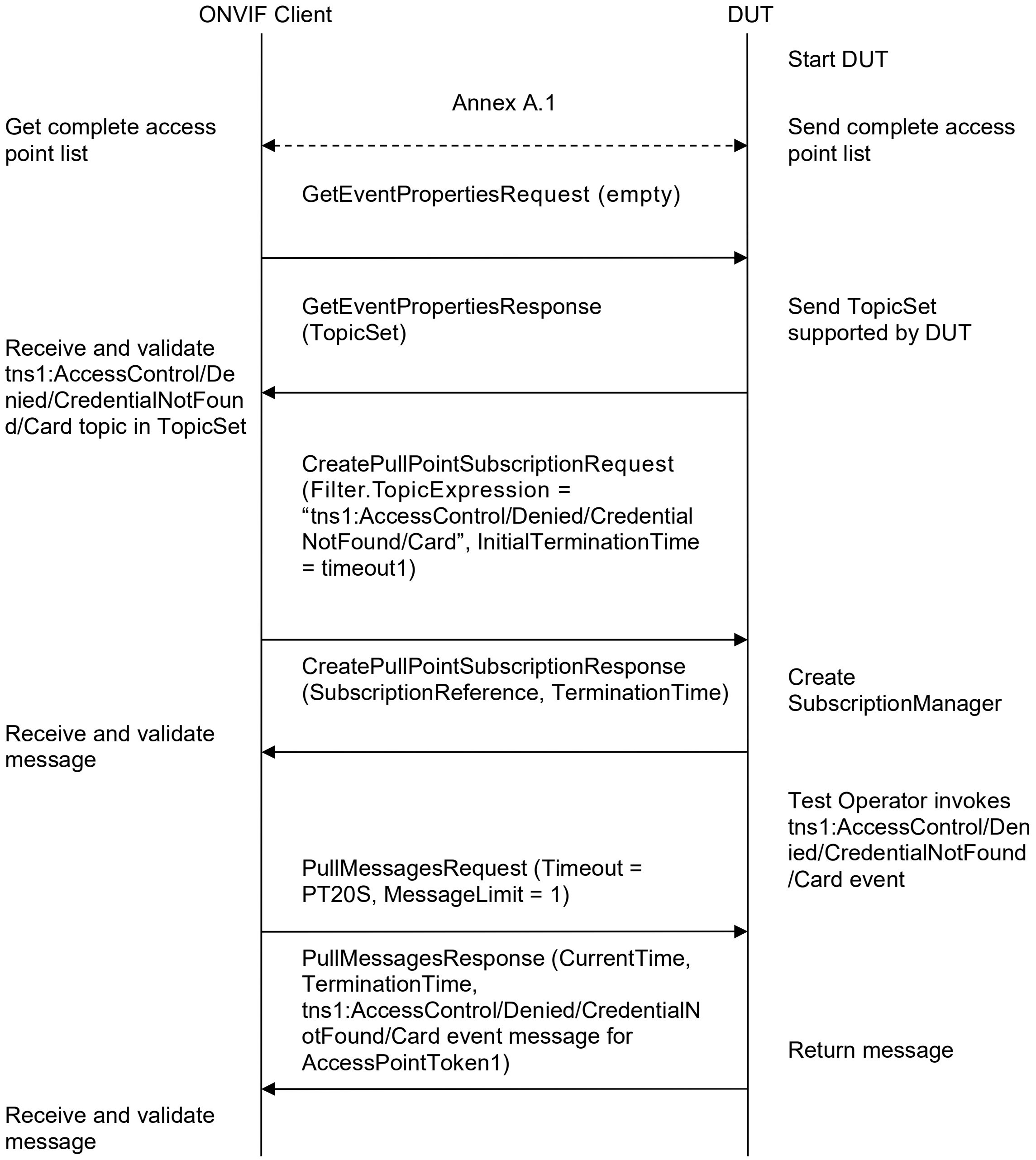
WSDL Reference: event.wsdl, accesscontrol.wsdl

Test Purpose: To verify tns1:AccessControl/Denied/CredentialNotFound/Card generation and to verify tns1:AccessControl/Denied/CredentialNotFound/Card format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with tns1:AccessControl/Denied/CredentialNotFound/Card event capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/Denied/CredentialNotFound/Card topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/Denied/CredentialNotFound/Card”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/Denied/CredentialNotFound/Card event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Denied/CredentialNotFound/Card event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get complete list of access points from the DUT (see Annex A.1).

4. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

5. Verify the GetEventPropertiesResponse message from the DUT.

6. Check if there is an event with Topic tns1:AccessControl/Denied/CredentialNotFound/Card. If there is no event with such Topic fail the test and skip other steps.

7. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

8. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

9. Check that this event contains Data.SimpleItemDescription item with Name = "Card" and Type = " xs:string".

10. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/Denied/CredentialNotFound/Card Topic as Filter and an InitialTerminationTime of timeout1.

11. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

12. Test Operator will invoke tns1:AccessControl/Denied/CredentialNotFound/Card event for any Access Points.

13. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

14. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 13.

15. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

16. Verify that TopicExpression is equal to tns1:AccessControl/Denied/CredentialNotFound/Card for the Notification message.

17. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to one of existing Access Point Tokens (e.g. complete list of access points contains Access Point with the same token).

18. Verify that the notification contains Data.SimpleItem item with Name = "Card" and Value with type is equal to xs:string.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/Denied/CredentialNotFound/Card event.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value, invalid Card).

The DUT did not return at least one Access Point on step 3.

The DUT did not return valid Topic tns1:AccessControl/Denied/CredentialNotFound/Card in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will be failed, if no Notification message is received for step 14 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.10 Duress events

4.10.1 ACCESS CONTROL – DURESS

Test Label: Access Control Service Events Check (Duress).

Test Case ID: ACCESSCONTROL-10-1-2

ONVIF Core Specification Coverage: tns:AccessControl/Duress (Access Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: none

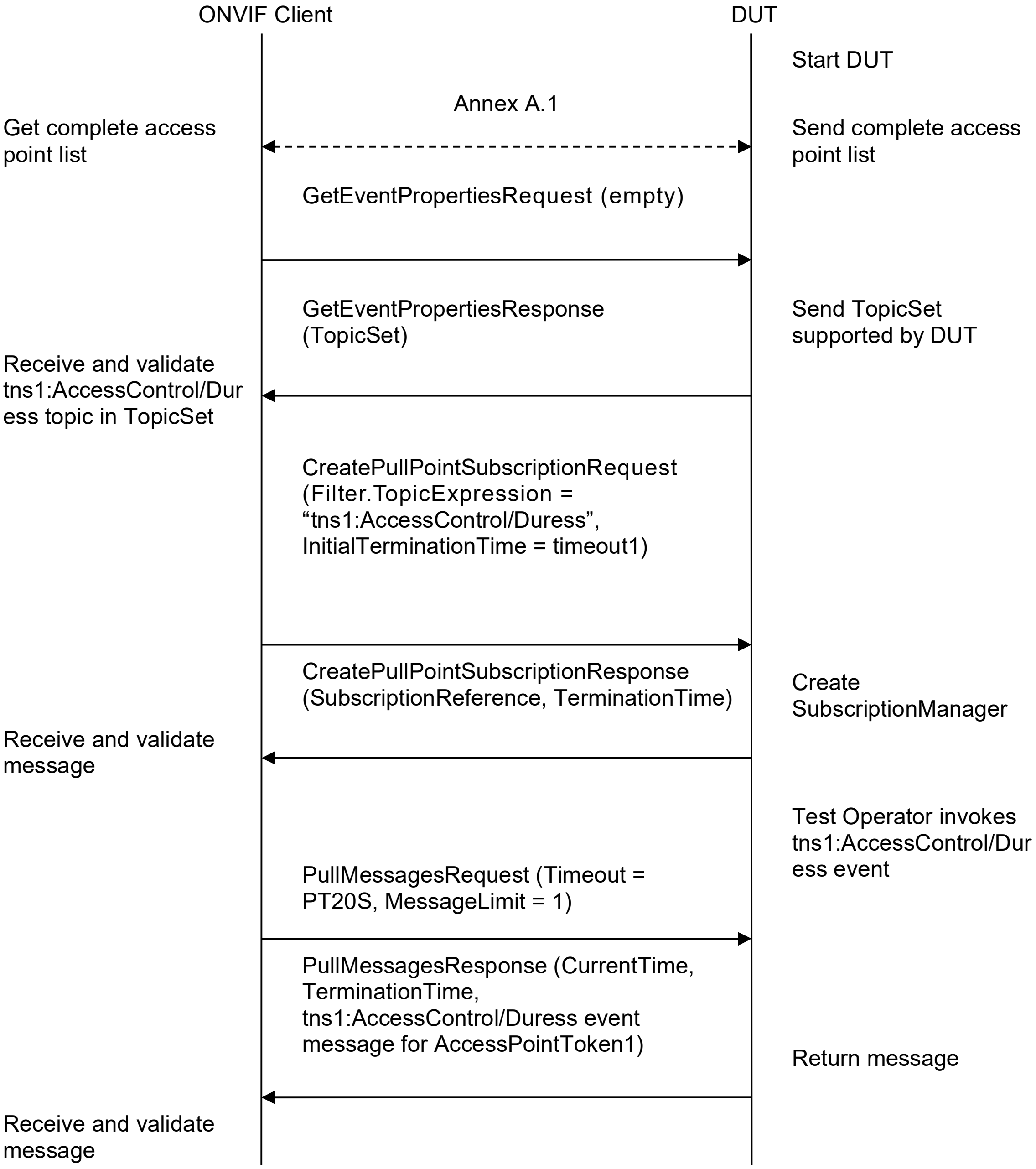
WSDL Reference: event.wsdl, accesscontrol.wsdl

Test Purpose: To verify tns1:AccessControl/Duress event generation and to verify tns1:AccessControl/Duress event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point with Duress capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:AccessControl/Duress topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:AccessControl/Duress”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:AccessControl/Duress event
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:AccessControl/Duress event message for AccessPointToken1)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of access points from the DUT (see Annex A.1).

4. If there is no Access Point with Capabilities.Duress = "true" on complete list of access points, fail the test and skip other steps.

5. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

6. Verify the GetEventPropertiesResponse message from the DUT.

7. Check if there is an event with Topic tns1:AccessControl/Duress. If there is no event with such Topic, skip other steps, fail the test and go to the next test.

8. Check that this event is not a Property event (MessageDescription.IsProperty = "false").

9. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

10. Check that if this event contains Data.SimpleItemDescription item with Name = "CredentialToken", then it has Type = "pt:ReferenceToken".

11. Check that if this event contains Data.SimpleItemDescription item with Name = "CredentialHolderName", then it has Type = " xs:string".

12. Check that this event contains Data.SimpleItemDescription item with Name = "Reason" and Type = "xs:string".

13. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:AccessControl/Duress Topic as Filter and an InitialTerminationTime of timeout1.

14. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

15. Test Operator will invoke tns1:AccessControl/Duress event for any Access Point with Capabilities.Duress = "true".

16. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

17. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. If no NotificationMessage were received repeat step 16.

18. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).

19. Verify that TopicExpression is equal to tns1:AccessControl/Duress for the Notification message.

20. Verify that the notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value is equal to one of existing Access Point Tokens with Capabilities.Duress = "true" (e.g. complete list of access points contains Access Point with the same token and this Access Point has Capabilities.Duress = "true").

21. Verify that the notification contains Data.SimpleItem item with Name = "Reason" and Value with type is equal to xs:string.

22. Verify that the notification which contains Data.SimpleItem item with Name = «CredentialToken» contains Value with type is equal to pt:ReferenceToken.

23. Verify that the notification which contains Data.SimpleItem item with Name = "CredentialHolderName" contains Value with type is equal to xs:string.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a PullMessagesResponse message that contains one tns1:AccessControl/Duress event.

The DUT sent an invalid Notification message (no corresponding Source.SimpleItem or Data.SimpleItem, wrong type of Value fields, invalid AccessPointToken value, invalid CredentialToken, invalid CredentialHolderName, invalid Reason).

The DUT did not return at least one Access Point at step 3.

The DUT did not return valid Topic tns1:AccessControl/Duress in GetEventPropertiesResponse.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

Note: Test will fail, if no Notification message is received at step 17 for a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.11 Consistency

4.11.1 GET AREA INFO LIST AND GET ACCESS POINT INFO LIST CONSISTENCY

Test Label: Get Access Point Info List and Get Area Info List Consistency Verification.

Test Case ID: ACCESSCONTROL-4-1-2

ONVIF Core Specification Coverage: GetAccessPointInfoList (ONVIF Access Control Service Specification), GetAreaInfoList (ONVIF Access Control Service Specification)

Command under test: GetAccessPointInfoList, GetAreaInfoList

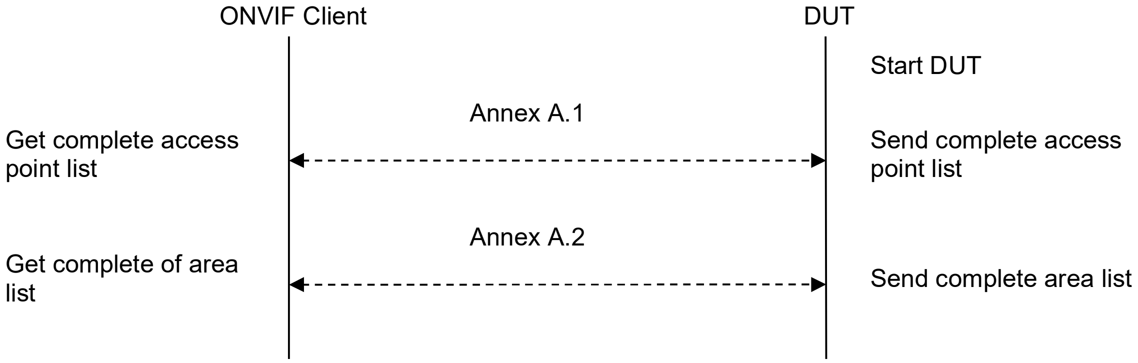
WSDL Reference: accesscontrol.wsdl

Test Purpose: To verify that Area Info List contains all Areas from Access Point Info List.

Pre-Requisite: Access Control Service was received from the DUT. At least one Access Point is configured and added to the DUT. At least one Area is configured and added to the DUT, if the DUT supports Areas.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
Annex A.2
Get complete of area list
Send complete area list

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of access points from the DUT (see Annex A.1)
4. Get a complete list of areas from the DUT (see Annex A.2)
5. Verify that the complete list of areas contains all Areas from AreaTo and AreaFrom elements of a complete access points list.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT returned at least one value in AreaTo or AreaFrom element of complete access points list that was not listed in the complete list of areas.

The DUT did not return at least one Access Point at step 3.

4.12 Access Point Configuration

4.12.1 ACCESS CONTROL – ADD OR CHANGE ACCESS POINT EVENT

Test Label: Access Control Service Add or Change Access Point Event Check.

Test Case ID: ACCESSCONTROL-12-1-1

ONVIF Core Specification Coverage: tns1:Configuration/AccessPoint/Changed (Access Control Service Specification), GetAccessPointInfo (ONVIF Access Control Service Specification)

Command under test: GetAccessPointInfo

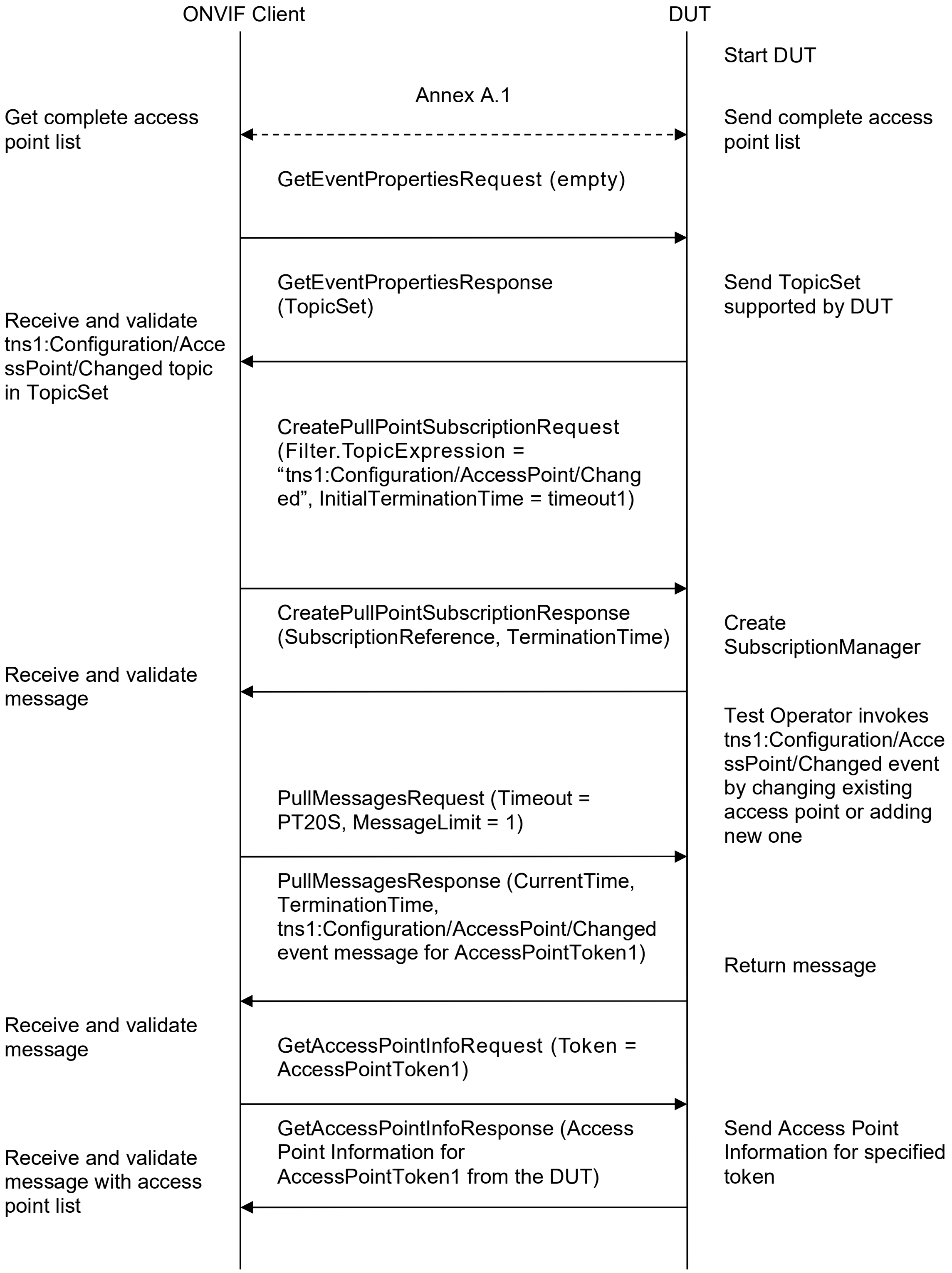
WSDL Reference: event.wsdl, accesscontrol.wsdl

Test Purpose: To verify tns1:Configuration/AccessPoint/Changed event generation after adding new access point or changing access point configuration to the DUT and to verify: tns1:Configuration/AccessPoint/Changed event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. The DUT allows adding or changing Access Points.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:Configuration/AccessPoint/Changed topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Configuration/AccessPoint/Changed”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:Configuration/AccessPoint/Changed event by changing existing access point or adding new one
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Configuration/AccessPoint/Changed event message for AccessPointToken1)
Return message
Receive and validate message
GetAccessPointInfoRequest (Token = AccessPointToken1)
GetAccessPointInfoResponse (Access Point Information for AccessPointToken1 from the DUT)
Send Access Point Information for specified token
Receive and validate message with access point list

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

4. Verify the GetEventPropertiesResponse message from the DUT.

5. Check if there is an event with Topic tns1:Configuration/AccessPoint/Changed. If there is no event with such Topic fail the test and skip other steps.

6. Check that this event isn’t a Property event (MessageDescription.IsProperty = "false").

7. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

8. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Configuration/AccessPoint/Changed Topic as Filter and an InitialTerminationTime of timeout1.

9. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

10. Test Operator will add Access Point or change Access Point configuration.

11. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

12. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. Repeat step 11 until Notification received.

13. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).

14. Verify that TopicExpression is equal to tns1:Configuration/AccessPoint/Changed for received message.

15. Verify that notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value with type is equal to tdc:ReferenceToken.

16. ONVIF Client will invoke GetAccessPointInfoRequest message (**Token** from Notification message) to retrieve subset of AccessPoint Information from the DUT.

17. Verify the GetAccessPointInfoResponse message from the DUT. Verify that requested Access Point was returned.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send PullMessagesResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a GetAccessPointInfoResponse with specified Access Point.

The DUT did not send a Notification message that contains an event tns1:Configuration/AccessPoint/Changed with valid AccessPointToken.

The DUT sent an invalid Notification message (invalid AccessPointToken value).

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

Note: ONVIF Client on step 12 will wait Notification messages until expected notification is received or Operation Delay has expired.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.12.2 ACCESS CONTROL – REMOVE ACCESS POINT EVENT

Test Label: Access Control Service Remove Access Point Event Check.

Test Case ID: ACCESSCONTROL-12-1-2

ONVIF Core Specification Coverage: tns1:Configuration/AccessPoint/Removed (Access Control Service Specification), GetAccessPointInfo (ONVIF Access Control Service Specification)

Command under test: GetAccessPointInfo

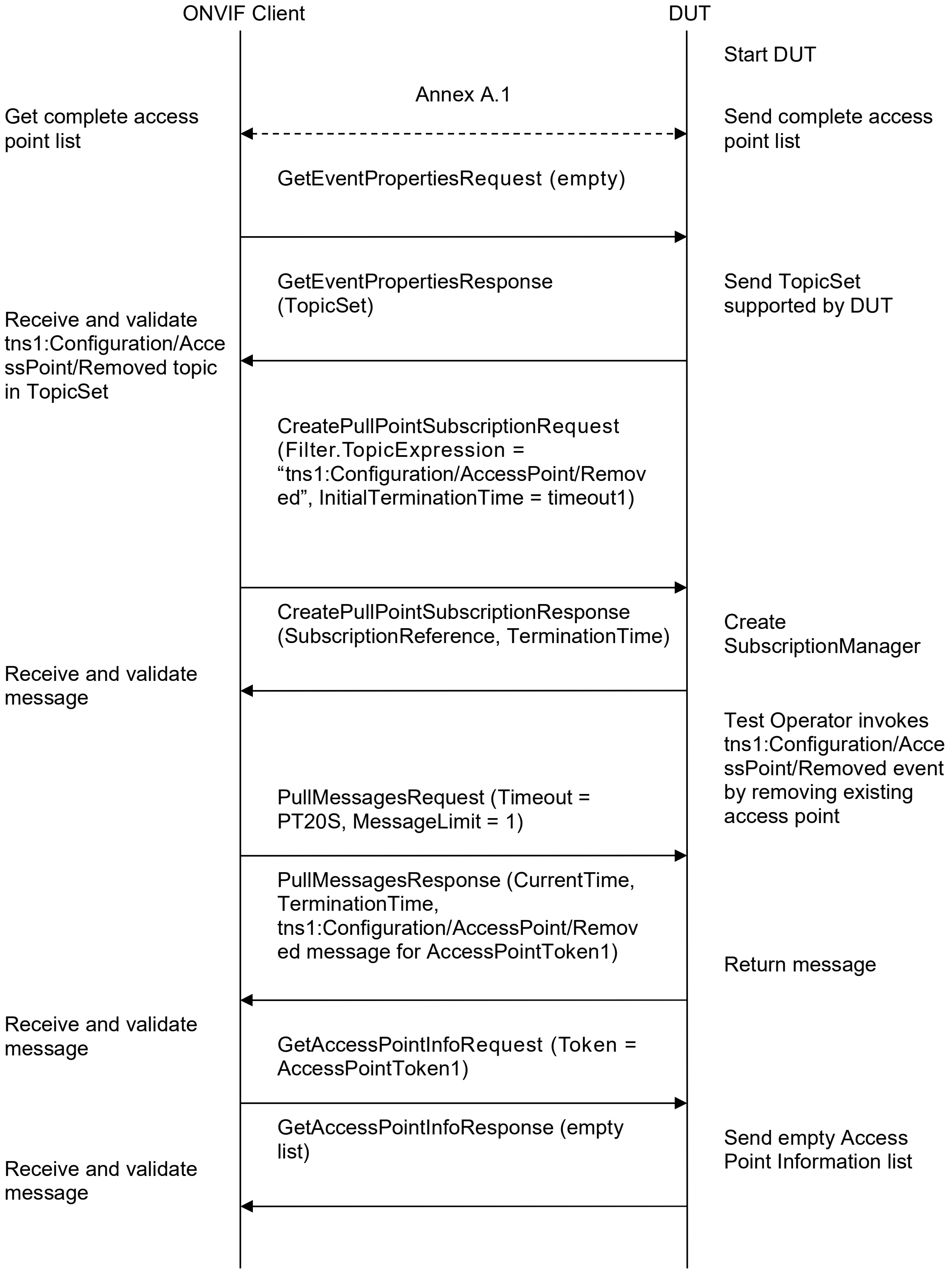
WSDL Reference: event.wsdl, accesscontrol.wsdl

Test Purpose: To verify tns1:Configuration/AccessPoint/Removed event generation after removing access point configuration to the DUT and to verify tns1:Configuration/AccessPoint/Removed event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Access Point is configured and added to the DUT. The DUT supports Access Points remove.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete access point list
Send complete access point list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:Configuration/AccessPoint/Removed topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Configuration/AccessPoint/Removed”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:Configuration/AccessPoint/Removed event by removing existing access point
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Configuration/AccessPoint/Removed message for AccessPointToken1)
Return message
Receive and validate message
GetAccessPointInfoRequest (Token = AccessPointToken1)
GetAccessPointInfoResponse (empty list)
Send empty Access Point Information list
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.

4. Verify the GetEventPropertiesResponse message from the DUT.

5. Check if there is an event with Topic tns1:Configuration/AccessPoint/Removed. If there is no event with such Topic, fail the test and skip other steps.

6. Check that this event isn’t a Property event (MessageDescription.IsProperty = "false").

7. Check that this event contains Source.SimpleItemDescription item with Name = "AccessPointToken" and Type = "pt:ReferenceToken".

8. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Configuration/AccessPoint/Removed Topic as Filter and an InitialTerminationTime of timeout1.

9. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

10. Test Operator will remove Access Point.

11. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

12. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. Repeat step 11 until Notification received.

13. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).

14. Verify that TopicExpression is equal to tns1:Configuration/AccessPoint/Removed for received message.

15. Verify that notification contains Source.SimpleItem item with Name = "AccessPointToken" and Value with type is equal to tdc:ReferenceToken.

16. ONVIF Client will invoke GetAccessPointInfoRequest message (**Token** from Notification message) to retrieve subset of Access Point Information from the DUT.

17. Verify the GetAccessPointInfoResponse message from the DUT. Check that empty list was returned.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a GetAccessPointInfoResponse with empty list.

The DUT did not send a Notification message that contains an event tns1:Configuration/AccessPoint/Removed with valid AccessPointToken.

The DUT sent an invalid Notification message (invalid AccessPointToken value).

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

Note: ONVIF Client on step 12 will wait Notification messages until expected notification is received or Operation Delay has expired.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.13 Area Configuration

4.13.1 ACCESS CONTROL – ADD OR CHANGE AREA EVENT

Test Label: Access Control Service Add or Change Area Event Check.

Test Case ID: ACCESSCONTROL-13-1-1

ONVIF Core Specification Coverage: tns1:Configuration/Area/Change (Access Control Service Specification), GetAreaInfo (ONVIF Access Control Service Specification)

Command under test: GetAreaInfo

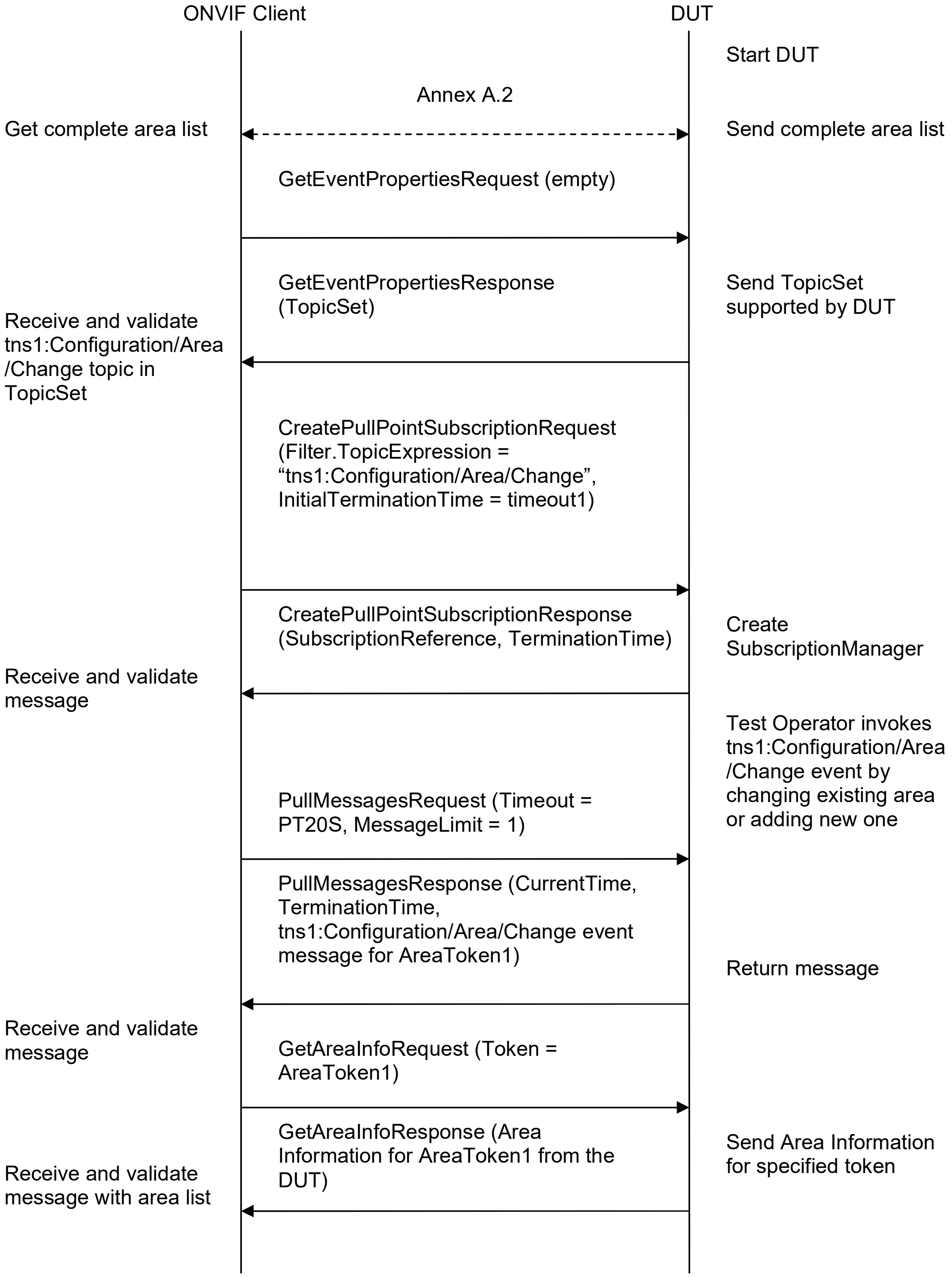
WSDL Reference: event.wsdl, accesscontrol.wsdl

Test Purpose: To verify: tns1:Configuration/Area/Change event generation after adding new area or changing area configuration to the DUT and to verify: tns1:Configuration/Area/Change event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. The DUT allows adding or changing Area.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete area list
Send complete area list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:Configuration/Area /Change topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Configuration/Area/Change”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:Configuration/Area /Change event by changing existing area or adding new one
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Configuration/Area/Change event message for AreaToken1)
Return message
Receive and validate message
GetAreaInfoRequest (Token = AreaToken1)
GetAreaInfoResponse (Area Information for AreaToken1 from the DUT)
Send Area Information for specified token
Receive and validate message with area list

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.
4. Verify the GetEventPropertiesResponse message from the DUT.
5. Check if there is an event with Topic tns1:Configuration/Area/Change. If there is no event with such Topic fail the test and skip other steps.
6. Check that this event isn’t a Property event (MessageDescription.IsProperty = "false").
7. Check that this event contains Source.SimpleItemDescription item with Name = "AreaToken" and Type = "pt:ReferenceToken".
8. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Configuration/Area/Changed Topic as Filter and an InitialTerminationTime of timeout1.
9. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
10. Test Operator will add Area or change Area configuration.
11. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
12. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. Repeat step 11 until Notification received.
13. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).
14. Verify that TopicExpression is equal to tns1:Configuration/Area/Change for received message.
15. Verify that notification contains Source.SimpleItem item with Name = "AreaToken" and Value with type is equal to tdc:ReferenceToken.
16. ONVIF Client will invoke GetAreaInfoRequest message (**Token** from Notification message) to retrieve subset of Area Information from the DUT.
17. Verify the GetAreaInfoResponse message from the DUT. Verify that requested Area was returned.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a GetAreaInfoResponse with specified Area.

The DUT did not send a Notification message that contains an event tns1:Configuration/Area/Change with valid AreaToken.

The DUT sent an invalid Notification message (invalid AreaToken value).

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

Note: ONVIF Client on step 12 will wait Notification messages until expected notification is received or Operation Delay after last notification has expired.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

4.13.2 ACCESS CONTROL – REMOVE AREA EVENT

Test Label: Access Control Service Remove Area Event Check.

Test Case ID: ACCESSCONTROL-13-1-2

ONVIF Core Specification Coverage: tns1:Configuration/Area/Removed (Access Control Service Specification), GetAreaInfo (ONVIF Access Control Service Specification)

Command under test: GetAreaInfo

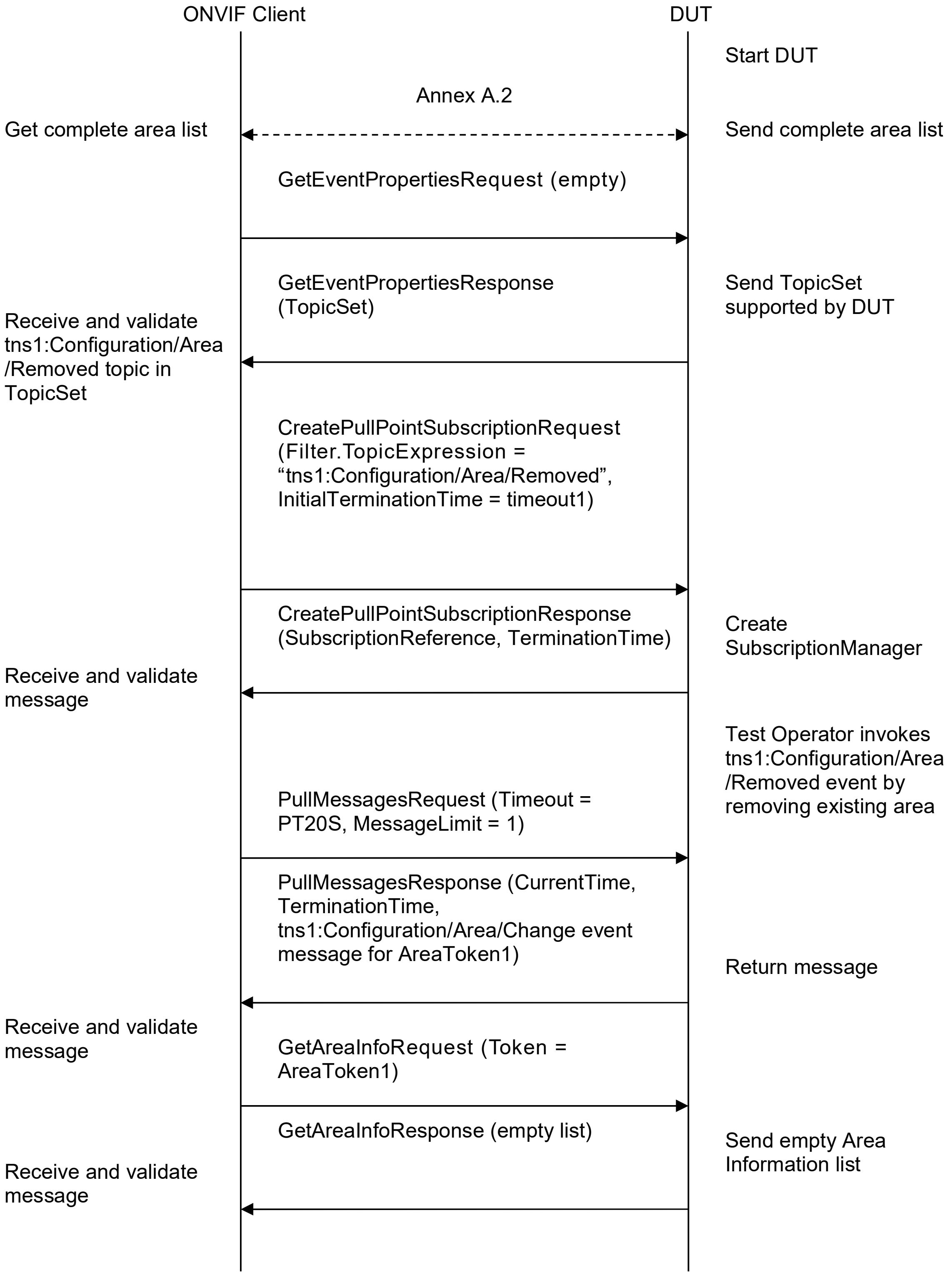
WSDL Reference: event.wsdl, accesscontrol.wsdl

Test Purpose: To verify tns1:Configuration/Area/Removed event generation after removing Area configuration to the DUT and to verify tns1:Configuration/Area/Removed event format.

Pre-Requisite: Event Service was received from the DUT. Access Control Service was received from the DUT. At least one Area is configured and added to the DUT. The DUT supports Area remove.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete area list
Send complete area list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:Configuration/Area /Removed topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Configuration/Area/Removed”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:Configuration/Area /Removed event by removing existing area
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Configuration/Area/Change event message for AreaToken1)
Return message
Receive and validate message
GetAreaInfoRequest (Token = AreaToken1)
GetAreaInfoResponse (empty list)
Send empty Area Information list
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke GetEventPropertiesRequest message to retrieve all events supported by the DUT.
4. Verify the GetEventPropertiesResponse message from the DUT.
5. Check if there is an event with Topic tns1:Configuration/Area/Removed. If there is no event with such Topic fail the test and skip other steps.
6. Check that this event isn’t a Property event (MessageDescription.IsProperty = "false").
7. Check that this event contains Source.SimpleItemDescription item with Name = "AreaToken" and Type = "pt:ReferenceToken".
8. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Configuration/Area/Removed Topic as Filter and an InitialTerminationTime of timeout1.
9. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
10. Test Operator will remove Area.
11. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
12. Verify that the DUT sends a PullMessagesResponse that contains one NotificationMessage. Repeat step 11 until Notification received.
13. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).
14. Verify that TopicExpression is equal to tns1:Configuration/Area/Removed for received message.
15. Verify that notification contains Source.SimpleItem item with Name = "AreaToken" and Value with type is equal to tdc:ReferenceToken.
16. ONVIF Client will invoke GetAreaInfoRequest message (Token from Notification message) to retrieve subset of Area Information from the DUT.
17. Verify the GetAreaInfoResponse message from the DUT. Check that empty list was returned.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a GetAreaInfoResponse with empty list.

The DUT did not send a Notification message that contains an event tns1:Configuration/Area/Removed with valid AreaToken.

The DUT sent an invalid Notification message (invalid AreaToken value).

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

Note: ONVIF Client on step 12 will wait Notification messages until expected notification is received or Operation Delay after last notification has expired.

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

Note: The Renew has to be used for renew subscription during test, if InitialTerminationTime expires.

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

Note: If DUT cannot accept the set value to a TerminationTime, ONVIF Client retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

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

Annex A

This section describes the meaning of the following definitions. These definitions are used in the test case description.

A.1 Get Complete Access Point Info List

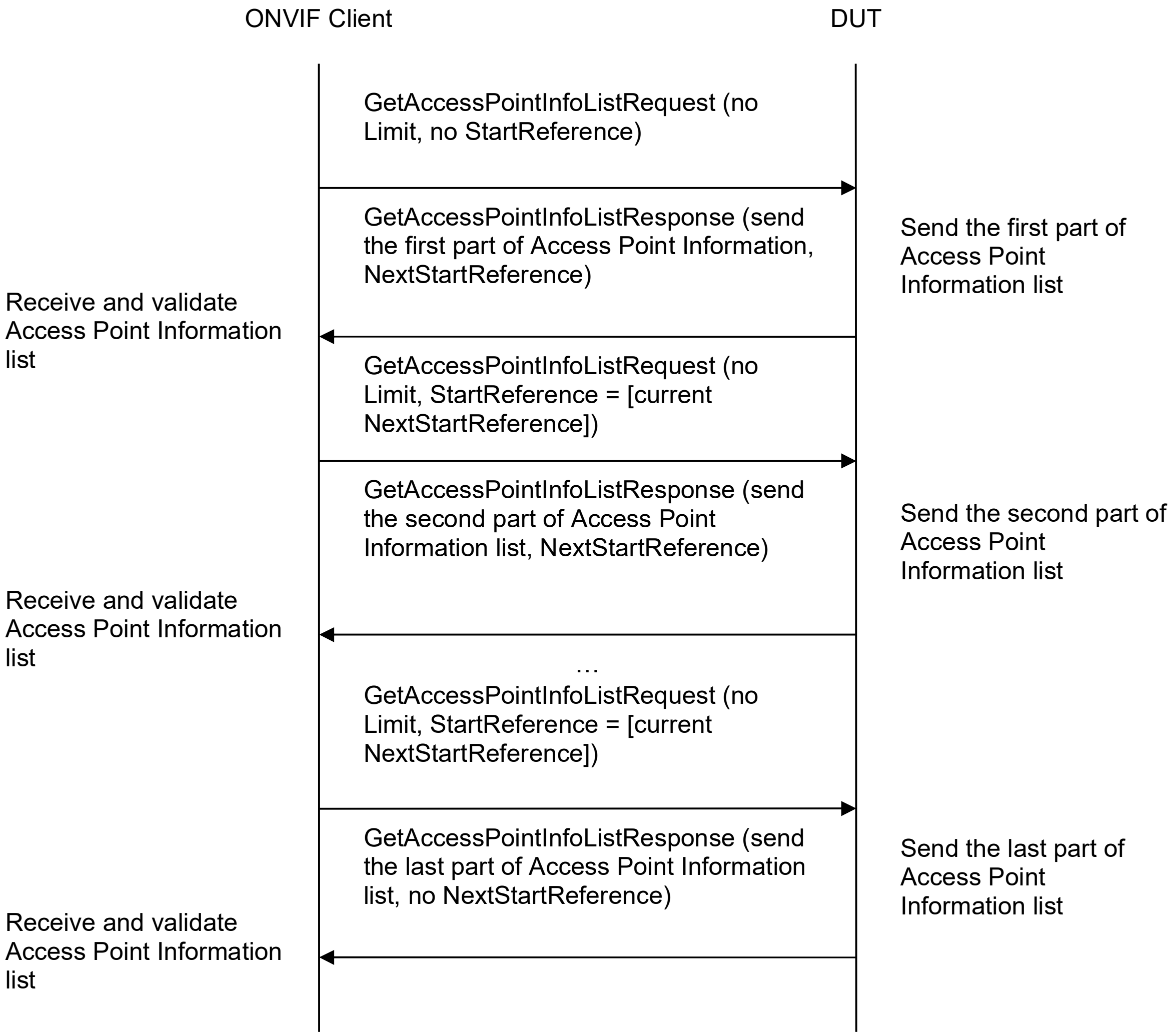
The following algorithm will be used to get a complete list of Access Points:

1. ONVIF Client will invoke GetAccessPointInfoListRequest message (no **Limit**, no StartReference) to retrieve the first part of Access Point Information list from the DUT.

2. Verify the GetAccessPointInfoListResponse message from the DUT.

3. If GetAccessPointInfoListResponse message contains NextStartReference, repeat steps 3-4 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting complete access point list.

The complete ordered list of access points with information will be made by the means of uniting all GetAccessPointInfoListResponse messages. Also the total number of access points will be calculated.


ONVIF Client
DUT
GetAccessPointInfoListRequest (no Limit, no StartReference)
GetAccessPointInfoListResponse (send the first part of Access Point Information, NextStartReference)
Send the first part of Access Point Information list
Receive and validate Access Point Information list
GetAccessPointInfoListRequest (no Limit, StartReference = [current NextStartReference])
GetAccessPointInfoListResponse (send the second part of Access Point Information list, NextStartReference)
Send the second part of Access Point Information list
Receive and validate Access Point Information list
…
GetAccessPointInfoListRequest (no Limit, StartReference = [current NextStartReference])
GetAccessPointInfoListResponse (send the last part of Access Point Information list, no NextStartReference)
Send the last part of Access Point Information list
Receive and validate Access Point Information list

A.2 Get Complete Area Info List

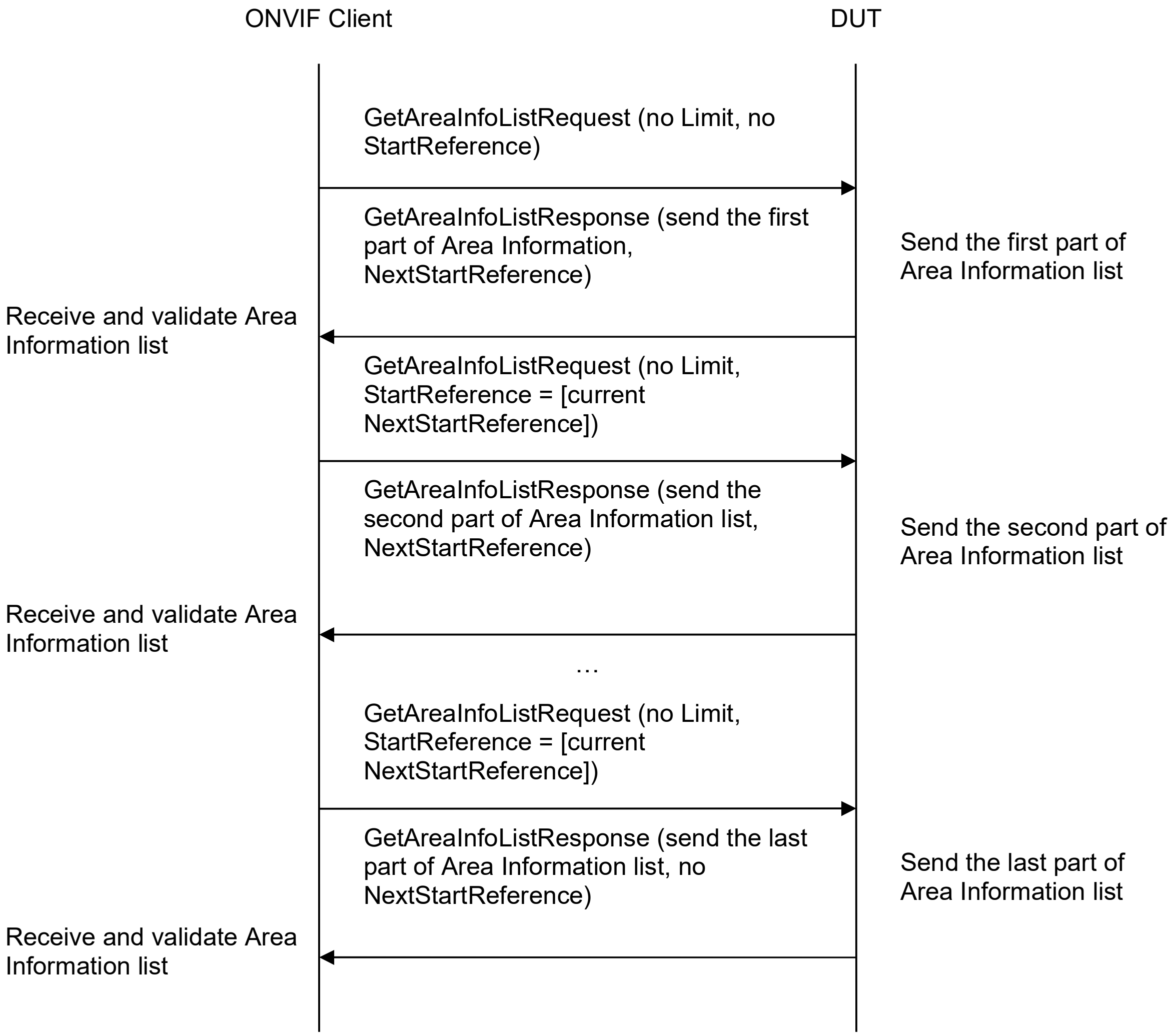
The following algorithm will be used to get a complete list of Areas:

1. ONVIF Client will invoke GetAreaInfoListRequest message (no **Limit**, no StartReference) to retrieve the first part of Area Information list from the DUT.

2. Verify the GetAreaInfoListResponse message from the DUT.

3. If GetAreaInfoListResponse message contains NextStartReference, repeat steps 3-4 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting complete area list.

The complete ordered list of areas with information will be made by the means of uniting all GetAreaInfoListResponse messages. Also the total number of areas will be calculated.


ONVIF Client
DUT
GetAreaInfoListRequest (no Limit, no StartReference)
GetAreaInfoListResponse (send the first part of Area Information, NextStartReference)
Send the first part of Area Information list
Receive and validate Area Information list
GetAreaInfoListRequest (no Limit, StartReference = [current NextStartReference])
GetAreaInfoListResponse (send the second part of Area Information list, NextStartReference)
Send the second part of Area Information list
Receive and validate Area Information list
…
GetAreaInfoListRequest (no Limit, StartReference = [current NextStartReference])
GetAreaInfoListResponse (send the last part of Area Information list, no NextStartReference)
Send the last part of Area Information list
Receive and validate Area Information list