Driving IP-based physical security through global standardization

ONVIF™

Door Control Test Specification

Version 14.06

June, 2014

Driving IP-based physical security through global standardization

© 2014 by ONVIF, Inc. All rights reserved.

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

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

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

Driving IP-based physical security through global standardization

Revision History

Ver.	Date	Description
13.06	Jun, 2013	First issue
13.12	Dec, 2013	Minor updates
14.06	Jun, 2014	The following tests were updated with Test Case ID change: ACCESS DOOR BLOCK DOOR DOUBLE LOCK DOOR LOCK DOOR UNLOCK DOOR LOCK DOWN DOOR LOCK DOWN RELEASE DOOR LOCK OPEN DOOR LOCK OPEN RELEASE DOOR

Driving IP-based physical security through global standardization

Table of Contents

ONVIF™ 1
Revision History 3
Table of Contents 4
1 Introduction 7
1.1 Scope 7
1.1.1 Capabilities 7
1.1.2 General 8
1.1.3 Door Control 8
1.1.4 Consistency 8
1.1.5 Property Events 8
1.1.6 Door Change Events 9
2 Terms and Definitions 10
2.1 Definitions 10
3 Test Overview 11
3.1 Test Setup 11
3.1.1 Network Configuration for device under test 11
3.2 Prerequisites 12
3.3 Test Policy 14
3.3.1 Capabilities 14
3.3.2 General 14
3.3.3 Door Control 15
3.3.4 Consistency 15
3.3.5 Property Events 16
3.3.6 Door Change Events 17
4 Door Control Test Cases 18
4.1 Capabilities 18
4.1.1 DOOR CONTROL SERVICE CAPABILITIES 18
4.1.2 GET SERVICES AND GET DOOR CONTROL SERVICE CAPABILITIES CONSISTENCY 19
4.2 General 20
4.2.1 GET DOOR STATE 20
4.2.2 GET DOOR STATE WITH INVALID TOKEN 22
4.2.3 GET DOOR INFO 23
4.2.4 GET DOOR INFO WITH INVALID TOKEN 26
4.2.5 GET DOOR INFO LIST – LIMIT 28
4.2.6 GET DOOR INFO LIST – START REFERENCE AND LIMIT 31
4.2.7 GET DOOR INFO LIST – NO LIMIT 35
4.2.8 GET DOOR INFO – TOO MANY ITEMS 37
4.3 Door Control 39

4.3.1 ACCESS DOOR WITH INVALID TOKEN 39
4.3.2 BLOCK DOOR WITH INVALID TOKEN 40
4.3.3 DOUBLE LOCK DOOR WITH INVALID TOKEN 41
4.3.4 LOCK DOOR WITH INVALID TOKEN 42
4.3.5 UNLOCK DOOR WITH INVALID TOKEN 43
4.3.6 LOCK DOWN DOOR WITH INVALID TOKEN 44
4.3.7 LOCK DOWN RELEASE DOOR WITH INVALID TOKEN 45
4.3.8 LOCK OPEN DOOR WITH INVALID TOKEN 46
4.3.9 LOCK OPEN RELEASE DOOR WITH INVALID TOKEN 47
4.3.10 ACCESS DOOR – COMMAND NOT SUPPORTED 48
4.3.11 BLOCK DOOR – COMMAND NOT SUPPORTED 50
4.3.12 DOUBLE LOCK DOOR – COMMAND NOT SUPPORTED 52
4.3.13 LOCK DOOR – COMMAND NOT SUPPORTED 54
4.3.14 UNLOCK DOOR – COMMAND NOT SUPPORTED 56
4.3.15 LOCK DOWN DOOR – COMMAND NOT SUPPORTED 58
4.3.16 LOCK DOWN RELEASE DOOR – COMMAND NOT SUPPORTED 60
4.3.17 LOCK OPEN DOOR – COMMAND NOT SUPPORTED 62
4.3.18 LOCK OPEN RELEASE DOOR – COMMAND NOT SUPPORTED 64
4.3.19 ACCESS DOOR 66
4.3.20 BLOCK DOOR 70
4.3.21 DOUBLE LOCK DOOR 73
4.3.22 LOCK DOOR 76
4.3.23 UNLOCK DOOR 81
4.3.24 LOCK OPEN DOOR 84
4.3.25 LOCK OPEN RELEASE DOOR 87
4.3.26 LOCK DOWN DOOR 91
4.3.27 LOCK DOWN RELEASE DOOR 95
4.4 Consistency 99
4.4.1 GET ACCESS POINT INFO LIST AND GET DOOR INFO LIST CONSISTENCY 99
4.5 Property Events 101
4.5.1 DOOR CONTROL – DOOR MODE EVENT 101
4.5.2 DOOR CONTROL – DOOR PHYSICAL STATE EVENT 105
4.5.3 DOOR CONTROL – DOOR PHYSICAL STATE EVENT STATE CHANGE 109
4.5.4 DOOR CONTROL – DOUBLE LOCK PHYSICAL STATE EVENT 112
4.5.5 DOOR CONTROL – DOUBLE LOCK PHYSICAL STATE EVENT STATE CHANGE 116
4.5.6 DOOR CONTROL – LOCK PHYSICAL STATE EVENT 119
4.5.7 DOOR CONTROL – LOCK PHYSICAL STATE EVENT STATE CHANGE 123
4.5.8 DOOR CONTROL – DOOR TAMPER EVENT 126
4.5.9 DOOR CONTROL – DOOR TAMPER EVENT STATE CHANGE 130
4.5.10 DOOR CONTROL – DOOR ALARM EVENT 133

4.5.11 DOOR CONTROL – DOOR ALARM EVENT STATE CHANGE 137
4.5.12 DOOR CONTROL – DOOR FAULT EVENT 140
4.5.13 DOOR CONTROL – DOOR FAULT EVENT STATE CHANGE 144
4.6 Door Configuration 147
4.6.1 DOOR CONTROL – ADD OR CHANGE DOOR EVENT 147
4.6.2 DOOR CONTROL – REMOVE DOOR EVENT 150
Annex A 154
A.1 Get Complete Door Info List 154
A.2 Get Complete Access Point Info List 155
A.3 Catching of Initialized event 156
A.4 User Interaction Process 156
A.5 Door's transition to Locked state 157

1 Introduction

The goal of the ONVIF test specification set is to make it possible to realize fully interoperable IP physical security implementation from different vendors. The set of ONVIF test specification describes the test cases need to verify the [ONVIF Network Interface Specs] and [ONVIF Conformance] requirements. 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 Door Control Test Specification acts as a supplementary document to the [ONVIF Network Interface Specs], illustrating test cases need 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 Door Control Test Specification defines and regulates the conformance testing procedure for the ONVIF conformant devices. Conformance testing is meant to be functional black-box testing. The objective of this specification is to provide the test cases to test individual requirements of ONVIF devices according to ONVIF Door Control service which is defined in [ONVIF Network Interface Specs].

The principal intended 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 Door Control Test Specification covers ONVIF Door Control service and Door 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 to get Door Control Service capabilities. It means that the following commands are covered by these test cases:

- GetServices (Device Management Service);
- GetServiceCapabilities.

1.1.2 General

General test cases cover verification to get Door list, information and state. It means that the following commands are covered by these test cases:

- GetDoorInfo;
- GetDoorInfoList;
- GetDoorState.

1.1.3 Door Control

Door Control test cases cover verification of door control commands and door state change. It means that the following commands and events are covered by these test cases:

- tns1:Door/State/DoorMode;
- AccessDoor;
- BlockDoor;
- DoubleLockDoor;
- LockDoor;
- UnlockDoor;
- LockDownDoor;
- LockDownReleaseDoor;
- LockOpenDoor;
- LockOpenReleaseDoor.

1.1.4 Consistency

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

- Access Point Info and Door Info.

1.1.5 Property Events

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

- tns1:Door/State/DoorMode;
- tns1:Door/State/DoorPhysicalState;
- tns1:Door/State/LockPhysicalState;
- tns1:Door/State/DoubleLockPhysicalState;

- tns1:Door/State/DoorAlarm;
- tns1:Door/State/DoorTamper;
- tns1:Door/State/DoorFault.

1.1.6 Door Change Events

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

- tns1:Configuration/Door/Change;
- tns1:Configuration/Door/Removed.

2 Terms and Definitions

2.1 Definitions

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

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 sensor.
Door Alarm	An abnormal state of the door where door is forced open or held open beyond the permitted time duration
Door Lock	A device that secures a door to prevent access, except when explicitly allowed by the access control system. Lock types include electromagnet, electric strike, etc.
Door Mode	Logical state of the door indicating whether the door is locked, unlocked, blocked, locked down or locked open etc.
Door Monitor	Also known as a Door Contact Sensor
Lock	An operation after which a door is locked and alarm is unmasked.
Momentary Access	An operation which invokes the same logic as upon normal access being granted to a credential.
Tamper Detector	Mechanism commonly available for doors, access points and controllers to detect physical tamper
Unlock	An operation to allow a door to be freely used for passage without any door alarms being triggered.

3 Test Overview

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

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.

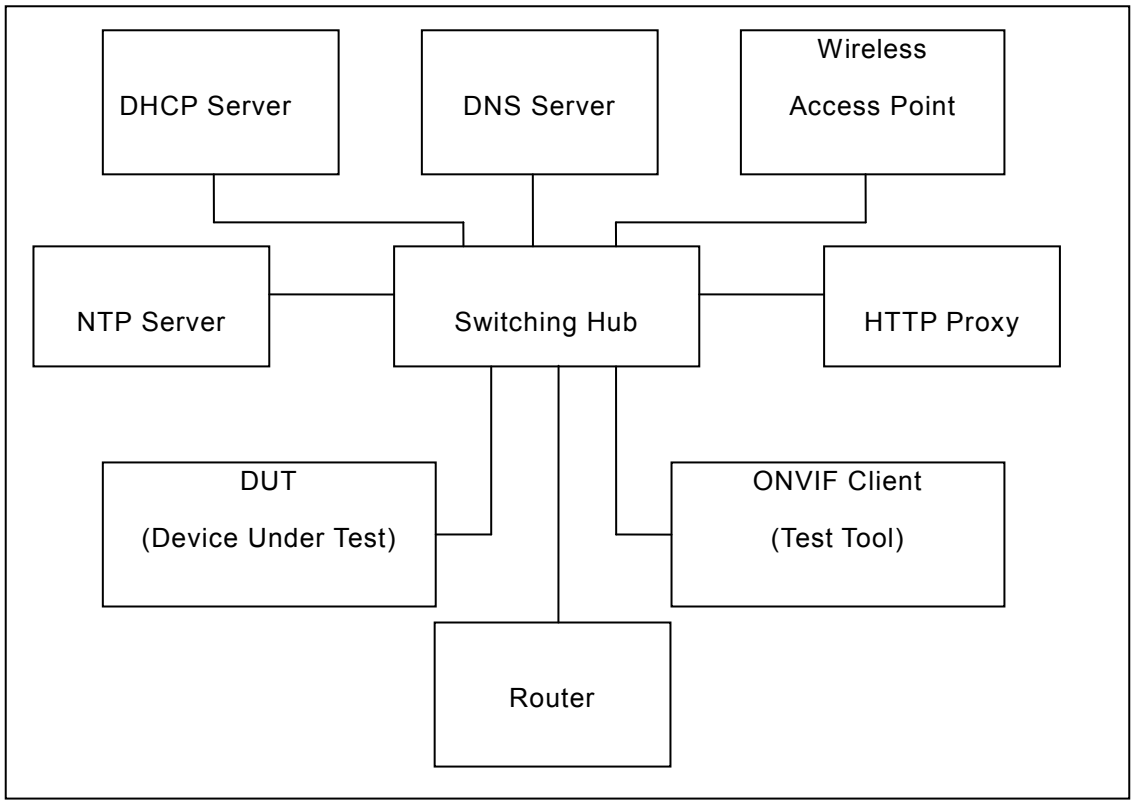


Figure 1: Test Configuration for DUT

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 behaviour of the DUT. It handles both expected and unexpected behaviour.

HTTP Proxy: provides facilitation in case of RTP and RTSP tunnelling 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 equipments 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 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):
  - 4.3.19 ACCESS DOOR
  - 4.3.20 BLOCK DOOR
  - 4.3.21 DOUBLE LOCK DOOR
  - 4.3.22 LOCK DOOR
  - 4.3.23 UNLOCK DOOR
  - 4.3.24 LOCK OPEN DOOR
  - 4.3.25 LOCK OPEN RELEASE DOOR
  - 4.3.26 LOCK DOWN DOOR
  - 4.3.27 LOCK DOWN RELEASE DOOR
  - 4.5.1 DOOR CONTROL – DOOR MODE EVENT
  - 4.5.2 DOOR CONTROL – DOOR PHYSICAL STATE EVENT
  - 4.5.3 DOOR CONTROL – DOOR PHYSICAL STATE EVENT STATE CHANGE
  - 4.5.4 DOOR CONTROL – DOUBLE LOCK PHYSICAL STATE EVENT
  - 4.5.5 DOOR CONTROL – DOUBLE LOCK PHYSICAL STATE EVENT STATE CHANGE

  - 4.5.6 DOOR CONTROL – LOCK PHYSICAL STATE EVENT
  - 4.5.7 DOOR CONTROL – LOCK PHYSICAL STATE EVENT STATE CHANGE
  - 4.5.8 DOOR CONTROL – DOOR TAMPER EVENT
  - 4.5.9 DOOR CONTROL – DOOR TAMPER EVENT STATE CHANGE
  - 4.5.10 DOOR CONTROL – DOOR ALARM EVENT
  - 4.5.11 DOOR CONTROL – DOOR ALARM EVENT STATE CHANGE
  - 4.5.12 DOOR CONTROL – DOOR FAULT EVENT
  - 4.5.13 DOOR CONTROL – DOOR FAULT EVENT STATE CHANGE
  - 4.6.1 DOOR CONTROL – ADD OR CHANGE DOOR EVENT
  - 4.6.2 DOOR CONTROL – REMOVE DOOR EVENT
- At least one Door is configured and added to the DUT.
- At least one Door with Access capability is configured and added to the DUT, if Access capability is supported by the DUT.
- At least one Door with Block capability is configured and added to the DUT, if Block capability is supported by the DUT.
- At least one Door with Double Lock capability is configured and added to the DUT, if Double Lock capability is supported by the DUT.
- At least one Door with Lock capability is configured and added to the DUT, if Lock capability is supported by the DUT.
- At least one Door with Unlock capability is configured and added to the DUT, if Unlock capability is supported by the DUT.
- At least one Door with Lock Down capability is configured and added to the DUT, if Lock Down capability is supported by the DUT.
- At least one Door with Lock Open capability is configured and added to the DUT, if Lock Open capability is supported by the DUT.
- At least one Door with Door Monitor capability is configured and added to the DUT, if Door Monitor capability is supported by the DUT.
- At least one Door with Lock Monitor capability is configured and added to the DUT, if Lock Monitor capability is supported by the DUT.
- At least one Door with Double Lock Monitor capability is configured and added to the DUT, if Double Lock Monitor capability is supported by the DUT.
- At least one Door with Tamper capability is configured and added to the DUT, if Tamper capability is supported by the DUT.
- At least one Door with Alarm capability is configured and added to the DUT, if Alarm capability is supported by the DUT.
- At least one Door with Fault capability is configured and added to the DUT, if Fault capability is supported by 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 Door 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 General

DUT shall give the Door 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 Door. And also at least one Door shall be added and configured on device.

DUT shall support the following commands:

- GetDoorInfo
- GetDoorInfoList
- GetDoorState

DUT shall not return any fault, if GetDoorInfo was invoked for non-exciting Door token. Such tokens shall be ignored.

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

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

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

DUT should return SOAP 1.2 fault message (InvalidArgVal/NotFound), if GetDoorState command was invoked for non-exciting Door token.

Please, refer to Section 4.2 for General Test Cases.

3.3.3 Door Control

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

DUT shall support GetServiceCapabilities command.

DUT shall support following property events:

- tns1:Door/State/DoorMode

DUT shall return capabilities for each specific door.

If DUT returns Access capability as supported by door, then DUT shall support AccessDoor command for this door. Otherwise, DUT should return SOAP 1.2 fault message.

If DUT returns Block capability as supported by door, then DUT shall support BlockDoor command for this door. Otherwise, DUT should return SOAP 1.2 fault message.

If DUT returns DoubleLock capability as supported by door, then DUT shall support DoubleLockDoor command for this door. Otherwise DUT should return SOAP 1.2 fault message.

If DUT returns Lock capability as supported by door, then DUT shall support LockDoor command for this door. Otherwise, DUT should return SOAP 1.2 fault message.

If DUT returns Unlock capability as supported by door, then DUT shall support UnlockDoor command for this door. Otherwise, DUT should return SOAP 1.2 fault message.

If DUT returns LockDown capability as supported by door, then DUT shall support LockDownDoor and LockDownReleaseDoor command for this door. Otherwise, DUT should return SOAP 1.2 fault message.

If DUT returns LockOpen capability as supported by door, then DUT shall support LockOpenDoor and LockOpenReleaseDoor command for this door. Otherwise, DUT should return SOAP 1.2 fault message.

DUT should return SOAP 1.2 fault message (InvalidArgVal/NotFound), if AccessDoor, BlockDoor, DoubleLockDoor, LockDoor, UnlockDoor, LockDownDoor, LockDownReleaseDoor, LockOpenDoor, or LockOpenReleaseDoor command was invoked for non-exciting Door token.

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 4.3 for Door Control Test Cases.

3.3.4 Consistency

DUT shall give the Door 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 a device.

DUT shall support at least one Door. And also at least one Door shall be added and configured on a device.

DUT shall support the following commands:

- GetAccessPointInfoList
- GetDoorInfoList
- DUT shall not return other Door tokens, than listed in GetAccessPointInfoList responses, in Entity field of GetAccessPointInfo's if Type is skipped or equal to tdc:Door.

Please, refer to Section 4.4 for Consistency Test Cases.

3.3.5 Property Events

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

DUT shall support GetServiceCapabilities command.

DUT shall support following property events:

- tns1:Door/State/DoorMode

DUT shall return capabilities for each specific door.

If DUT returns DoorMonitor capability as supported by door, then DUT shall support the following event for this door:

- tns1:Door/State/DoorPhysicalState

If DUT returns DoubleLockMonitor capability as supported by door, then DUT shall support the following event for this door:

- tns1:Door/State/DoubleLockPhysicalState

If DUT returns LockMonitor capability as supported by door, then DUT shall support the following event for this door:

- tns1:Door/State/LockPhysicalState

If DUT returns Tamper capability as supported by door, then DUT shall support the following event for this door:

- tns1:Door/State/DoorTamper

If DUT returns Alarm capability as supported by door, then DUT shall support the following event for this door:

- tns1:Door/State/DoorAlarm

If DUT returns Fault capability as supported by door, then DUT shall support the following event for this door:

- tns1:Door/State/DoorFault

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.

A test operator shall manually change the following properties if required:

- DoorPhysicalState
- LockPhysicalState
- DoubleLockPhysicalState
- Alarm
- Tamper
- Fault

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

3.3.6 Door Change Events

DUT shall give the Door Control Service entry point and Event Service entry point by GetServices command, if DUT supports Door 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 doors, then DUT shall support the following events:

- tns1:Configuration/Door/Change

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

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

- tns1:Configuration/Door/Removed

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

A test operator shall manually invoke following events if required:

- tns1:Configuration/Door/Change
- tns1:Configuration/Door/Removed

Please, refer to Section 4.6 for Door Change events Test Cases.

4 Door Control Test Cases

4.1 Capabilities

4.1.1 DOOR CONTROL SERVICE CAPABILITIES

Test Label: Door Control Capabilities Verification.

Test Case ID: DOORCONTROL-1-1-1

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

Command under test: GetServiceCapabilities (for Door Control Service)

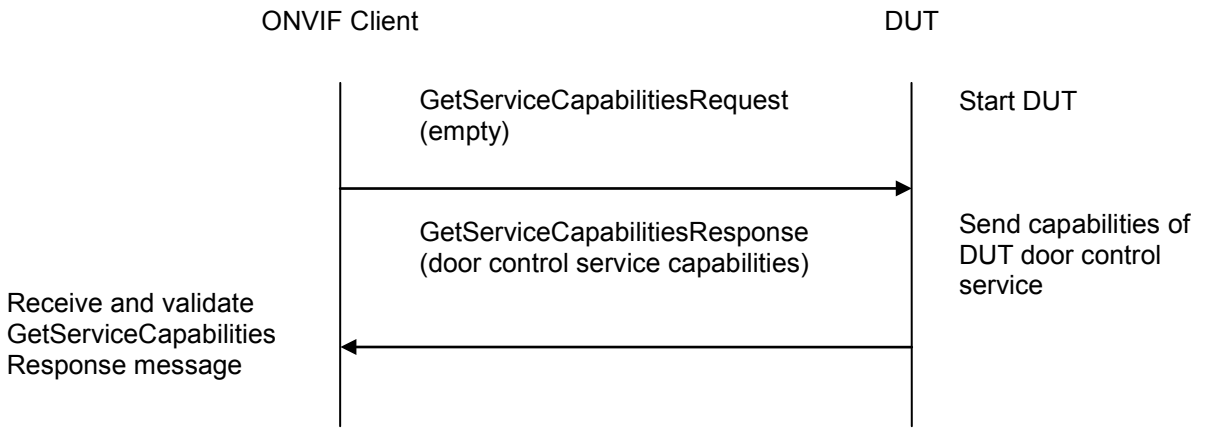
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify DUT Door Control Service Capabilities.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve door control service capabilities of the DUT.
4. Verify the GetServiceCapabilitiesResponse from the DUT. Verify that MaxLimit is greater than zero.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetServiceCapabilitiesResponse.

The DUT returned MaxLimit less than zero.

4.1.2 GET SERVICES AND GET DOOR CONTROL SERVICE CAPABILITIES CONSISTENCY

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

Test Case ID: DOORCONTROL-1-1-2

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

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

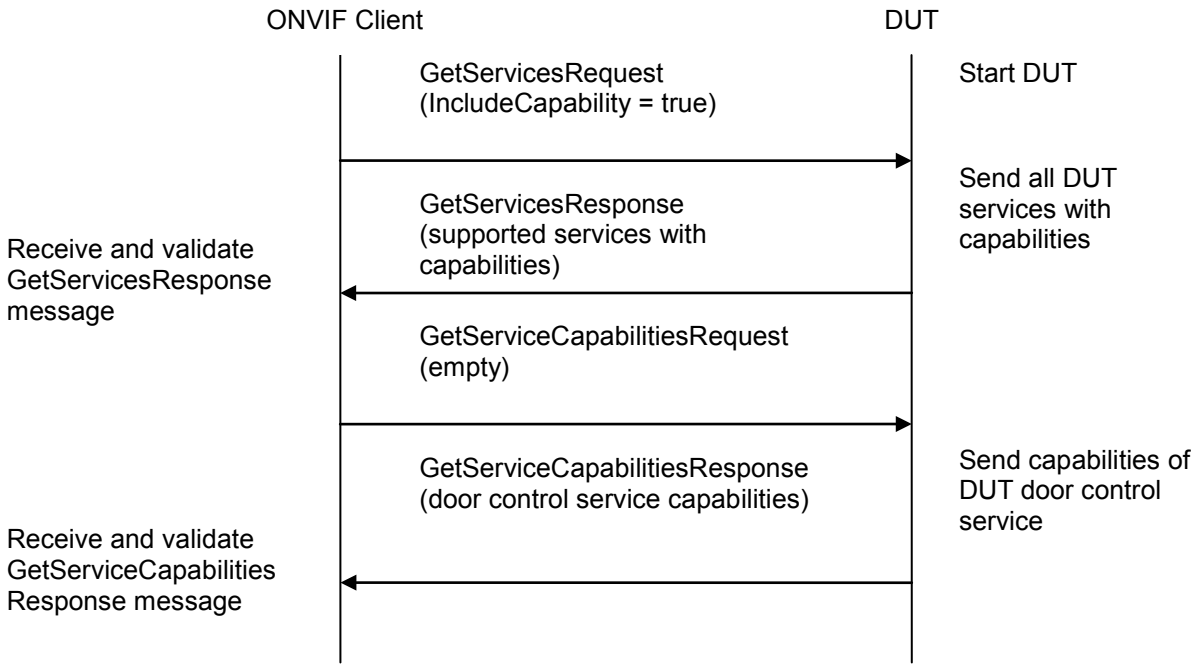
WSDL Reference: devicemgmt.wsdl, doorcontrol.wsdl

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

Pre-Requisite: None.

Test Configuration: ONVIF Client and DUT

Test Sequence:



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 Door Control service capabilities of the DUT.
6. Verify the GetServiceCapabilitiesResponse message from the DUT.

Test Result:

PASS –

The DUT passed 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 Door Controller service if it has Namespace element that is equal to “http://www.onvif.org/ver10/doorcontrol/wsdl”.

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

- MaxLimit attribute values are different.

4.2 General

4.2.1 GET DOOR STATE

Test Label: Get Door State Verification.

Test Case ID: DOORCONTROL-2-1-1

ONVIF Core Specification Coverage: GetDoorState (ONVIF Door Control Service Specification)

Command under test: GetDoorState

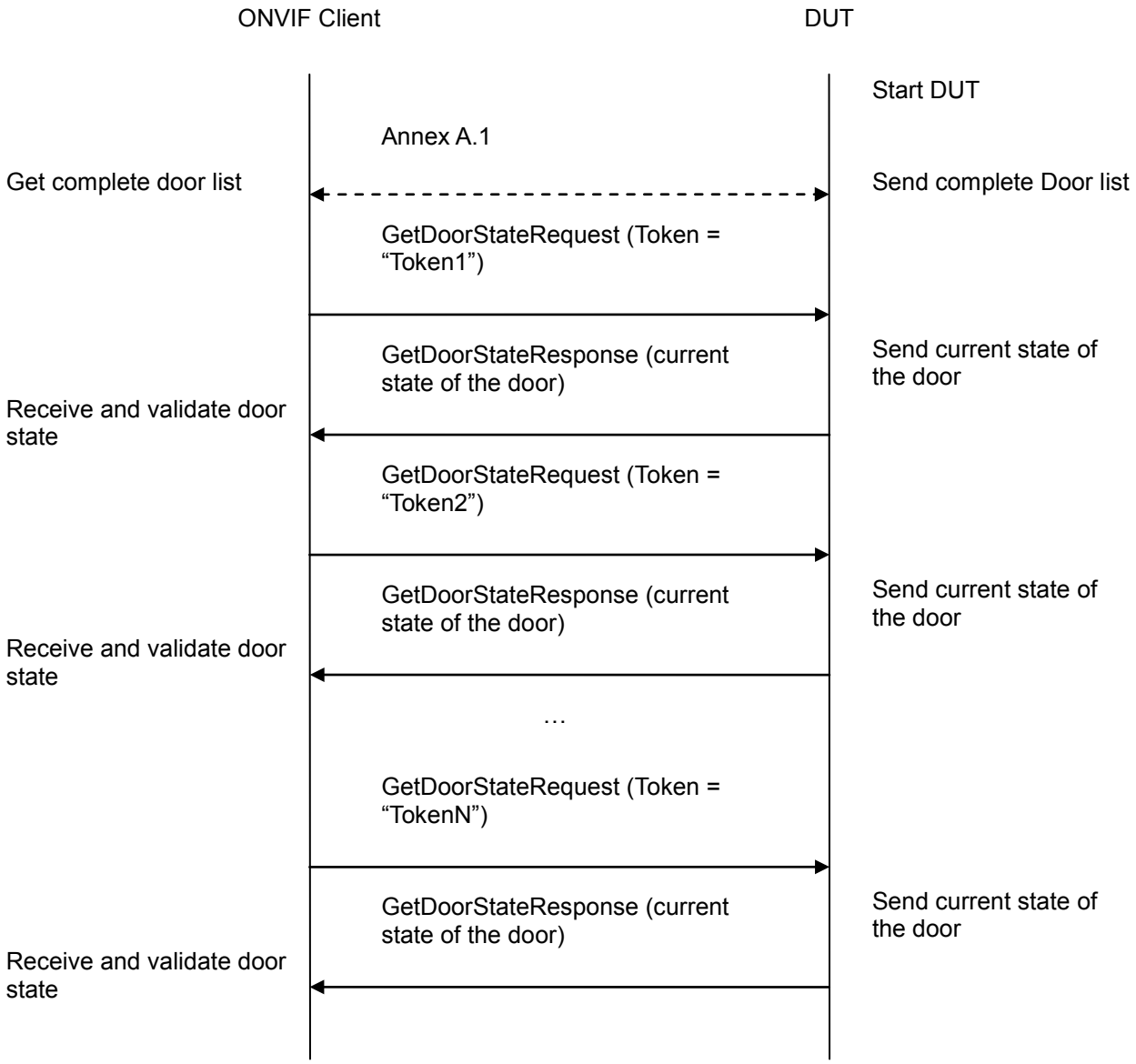
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify Get Door State.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. ONVIF Client will invoke GetDoorStateRequest message (**Token = “Token1”**, where Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) to retrieve current state of the door.
5. Verify the GetDoorStateResponse message from the DUT.
6. Check that if DoorInfo.Capabilities.DoorMonitor = "true", than DoorState.DoorPhysicalState is present. Otherwise, if DoorInfo.Capabilities.DoorMonitor = "false", then check that DoorState.DoorPhysicalState is skipped.

7. Check that if DoorInfo.Capabilities.LockMonitor = "true", than DoorState.LockPhysicalState is present. Otherwise, if DoorInfo.Capabilities.LockMonitor = "false", then check that DoorState.LockPhysicalState is skipped.

8. Check that if DoorInfo.Capabilities.DoubleLockMonitor = "true", than DoorState.DoubleLockPhysicalState is present. Otherwise, if DoorInfo.Capabilities.DoubleLockMonitor = "false", then check that DoorState.DoubleLockPhysicalState is skipped.

9. Check that if DoorInfo.Capabilities.Alarm = "true", than DoorState.Alarm is present. Otherwise if DoorInfo.Capabilities.Alarm = "false", then check that DoorState.Alarm is skipped.

10. Check that if DoorInfo.Capabilities.Tamper = "true", than DoorState.Tamper is present. Otherwise, if DoorInfo.Capabilities.Tamper = "false", than check that DoorState.Tamper is skipped.

11. Check that if DoorInfo.Capabilities.Fault = "true", than DoorState.Fault is present. Otherwise, if DoorInfo.Capabilities.Fault = "false", then check that DoorState.Fault is skipped.

12. Repeat steps 4-11 for all other tokens from the complete list of doors at step 3.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetDoorStateResponse message.

The DUT returned DoorState which contains unsupported monitors (DoorPhysicalState, LockPhysicalState, DoubleLockPhysicalState, Alarm, Tamper, and Fault).

The DUT returned DoorState which does not contain supported monitors (DoorPhysicalState, LockPhysicalState, DoubleLockPhysicalState, Alarm, Tamper, and Fault).

The DUT did not return at least one Door at step 3.

Note: If the DUT does not return any DoorInfo on a complete list of doors at step 3, skip steps 4-12, fail the test and go to the next test.

4.2.2 GET DOOR STATE WITH INVALID TOKEN

Test Label: Get Door State Verification with Invalid Token.

Test Case ID: DOORCONTROL-2-1-2

ONVIF Core Specification Coverage: GetDoorState (ONVIF Door Control Service Specification)

Command under test: GetDoorState

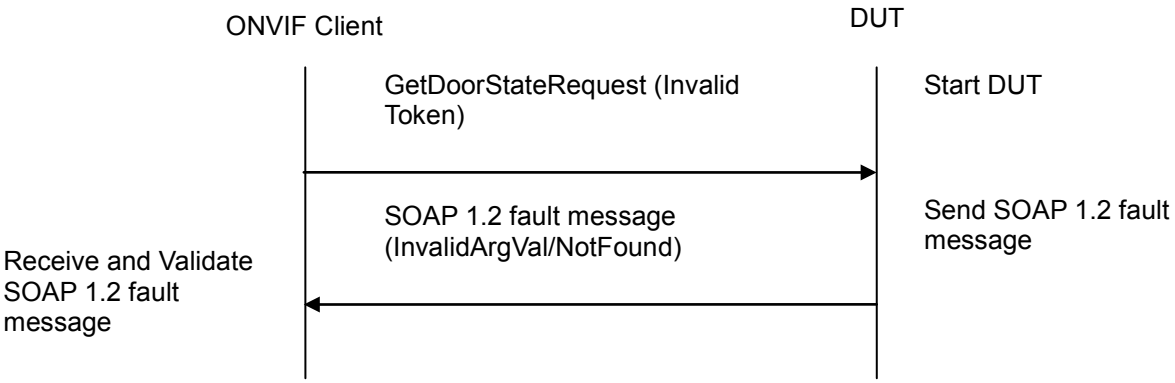
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify Get Door State List with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke GetDoorStateRequest 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.

4.2.3 GET DOOR INFO

Test Label: Get Door Verification.

Test Case ID: DOORCONTROL-2-1-3

ONVIF Core Specification Coverage: GetDoorInfo (ONVIF Door Control Service Specification)

Command under test: GetDoorInfo

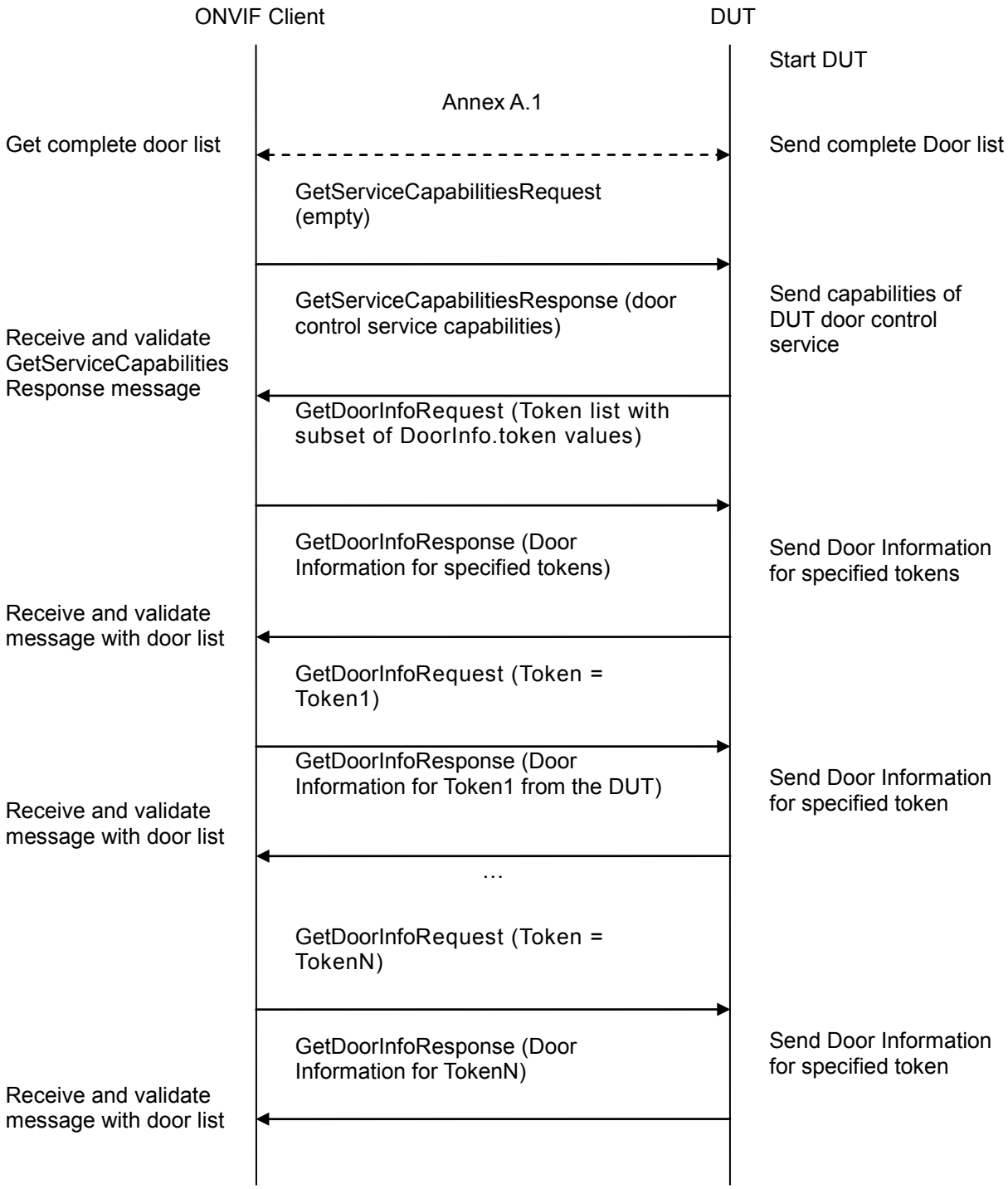
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify Get Door Info.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors 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 GetDoorInfoRequest message (**Token** list with subset of DoorInfo.token values from the complete list of doors at step 3 with token number equal to MaxLimit) to retrieve subset of Door Information from the DUT.
7. Verify the GetDoorInfoResponse message from the DUT.
8. Verify that all requested **DoorInfo** items are in GetDoorInfoResponse message.
9. ONVIF Client will invoke GetDoorInfoRequest message (**Token = Token1**, where Token1 is the first token from the complete list of doors at step 3) to retrieve Door Information for the specified token from the DUT.
10. Verify the GetDoorInfoResponse message from the DUT.
11. Verify that GetDoorInfoResponse message contains **DoorInfo** only for specified token.
12. Repeat steps 9-11 for all other tokens from the complete list of doors at step 3.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetDoorInfoResponse message.

The DUT returned GetDoorInfoResponse message that contains at least two DoorInfo items with tokens of the same value.

The DUT did not return requested DoorInfo items in GetDoorInfoResponse message for step 7 or 10.

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

The DUT did not send valid GetServiceCapabilitiesResponse.

The DUT did not return at least one Door at step 3.

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

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

4.2.4 GET DOOR INFO WITH INVALID TOKEN

Test Label: Get Door Info by Token List Verification with Invalid Token.

Test Case ID: DOORCONTROL-2-1-4

ONVIF Core Specification Coverage: GetDoorInfo (ONVIF Door Control Service Specification)

Command under test: GetDoorInfoList

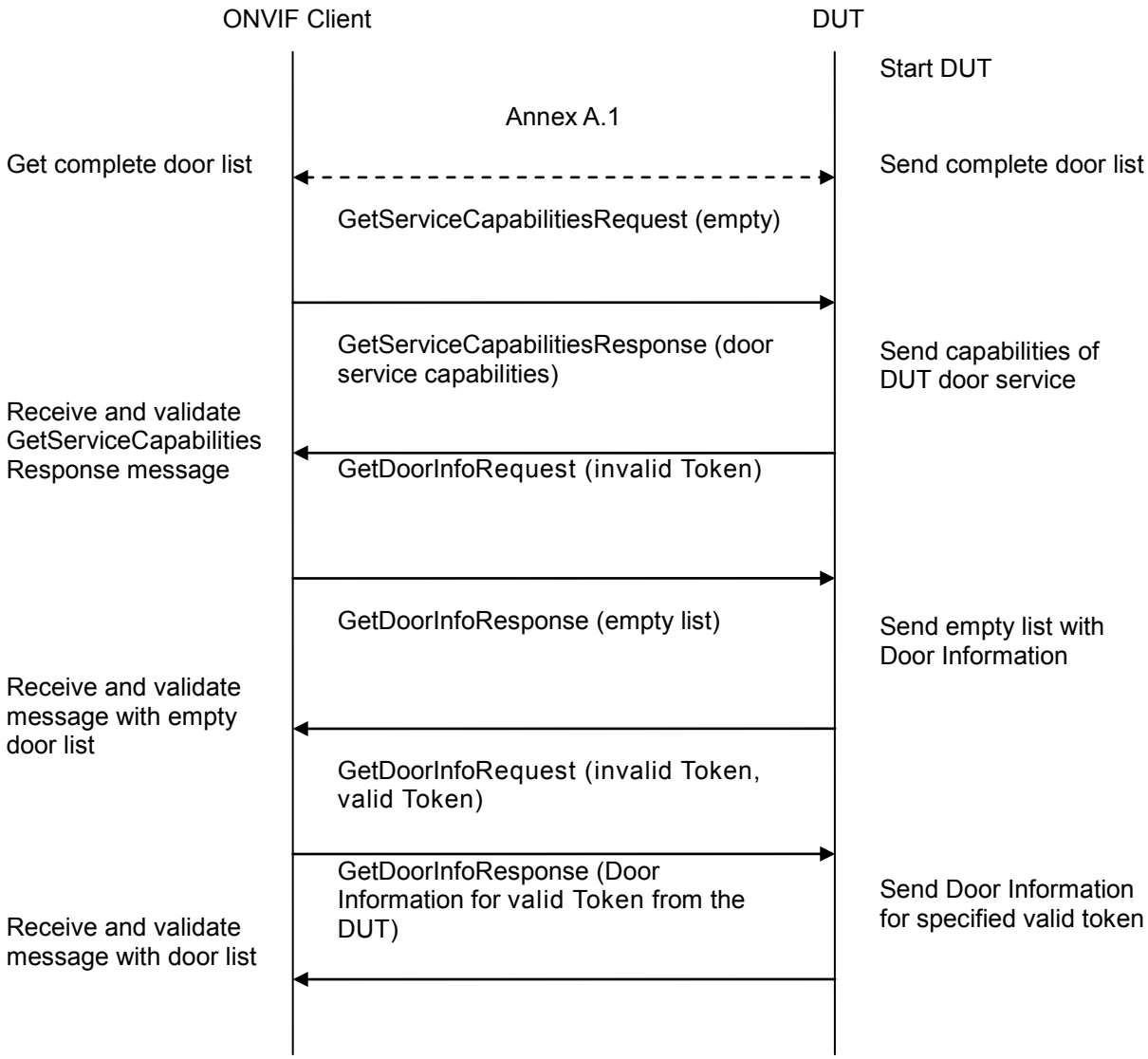
WSDL Reference: doorcontrol.wsdl

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

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get complete list of doors 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 GetDoorInfoRequest message **(invalid Token**).
7. Verify the GetDoorInfoResponse message from the DUT. Check that empty list was returned.
8. ONVIF Client will invoke GetDoorInfoRequest message **(invalid Token, valid Token**).

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

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetDoorInfoResponse message.

The DUT returned GetDoorInfoResponse message that contains at least two DoorInfo items with equal tokens.

The DUT did not send valid GetDoorInfoResponse message.

The DUT returned GetDoorInfoResponse message that contains any DoorInfo items for step 7.

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

The DUT did not return at least one Door at step 3.

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

4.2.5 GET DOOR INFO LIST – LIMIT

Test Label: Get Door Info List Verification with Limit.

Test Case ID: DOORCONTROL-2-1-6

ONVIF Core Specification Coverage: GetDoorInfoList (ONVIF Door Control Service Specification)

Command under test: GetDoorInfoList

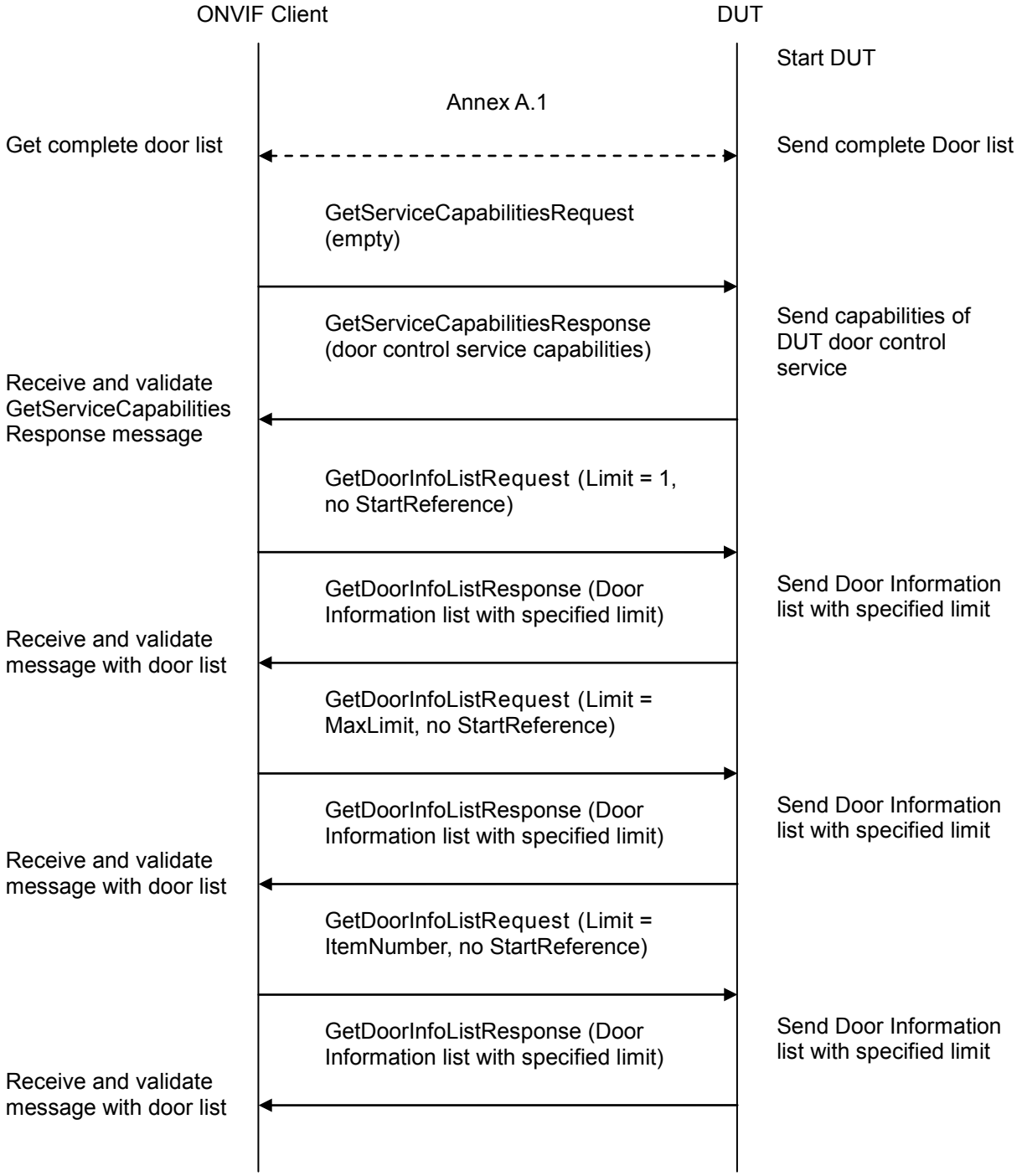
WSDL Reference: doorcontrol.wsdl

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

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors 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 GetDoorInfoListRequest message (**Limit = 1, no StartReference**) to retrieve the first item in the list of Door Information from the DUT.
7. Verify the GetDoorInfoListResponse message from the DUT.
8. Verify that GetDoorInfoListResponse message at step 7 does not contain number of items greater than 1. Verify that GetDoorInfoListResponse message at step 7 does not contain number of items greater than MaxLimit.
9. ONVIF Client will invoke GetDoorInfoListRequest message (**Limit = MaxLimit, no StartReference**) to retrieve the first part of Door Information list from the DUT and is limited by MaxLimit.
10. Verify the GetDoorInfoListResponse message from the DUT.
11. Verify that GetDoorInfoListResponse message at step 10 does not contain number of items greater than MaxLimit.
12. ONVIF Client will invoke GetDoorInfoListRequest message (**Limit = ItemNumber, no StartReference,** where ItemNumber is between 1 and minimal value of MaxLimit and the total number of doors) to retrieve Door Information sublist limited by ItemNumber from the DUT.
13. Verify the GetDoorInfoListResponse message from the DUT.
14. Verify that GetDoorInfoListResponse message at step 13 does not contain number of items greater than ItemNumber. Verify that GetDoorInfoListResponse 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 GetDoorInfoListResponse message.

The DUT returned more DoorInfo items in GetDoorInfoListResponse message than it was specified in Limit parameter.

The DUT returned more DoorInfo items in GetDoorInfoListResponse message than MaxLimit.

The DUT did not return at least one Door at step 3.

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

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

4.2.6 GET DOOR INFO LIST – START REFERENCE AND LIMIT

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

Test Case ID: DOORCONTROL-2-1-7

ONVIF Core Specification Coverage: GetDoorInfoList (ONVIF Door Control Service Specification)

Command under test: GetDoorInfoList

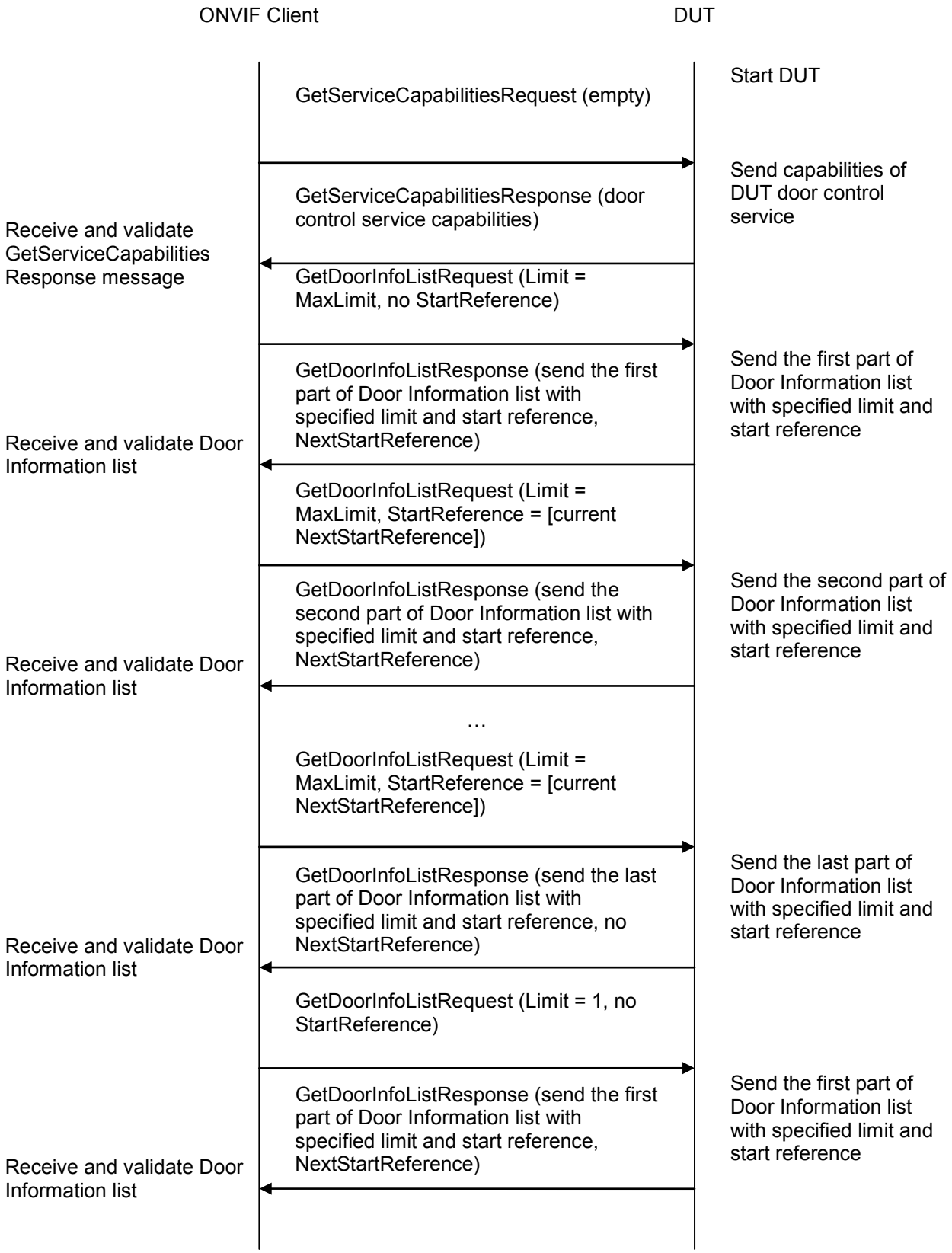
WSDL Reference: doorcontrol.wsdl

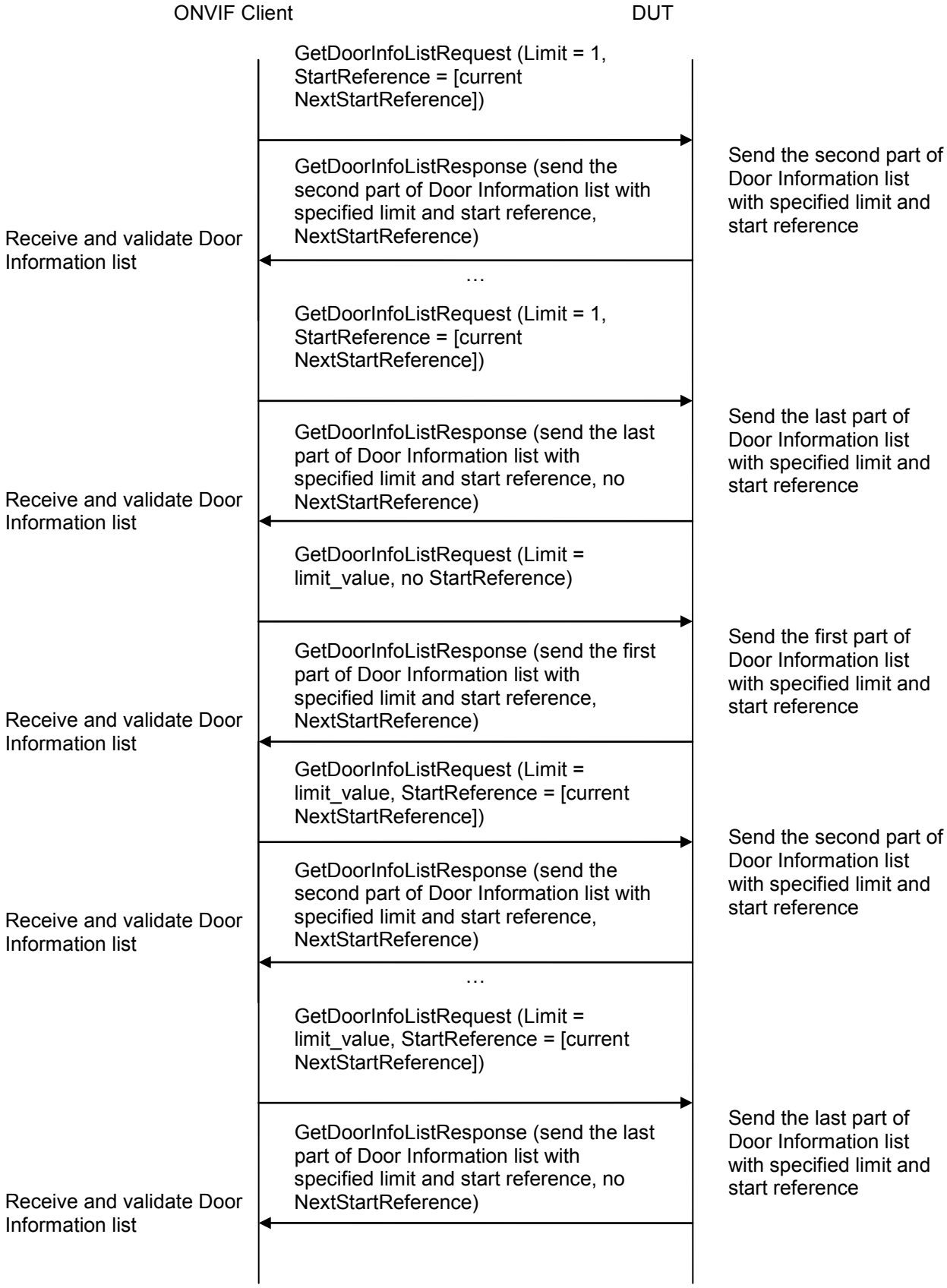
Test Purpose: To verify Get Door Info List using StartReference and Limit.

Pre-Requisite: Door Control Service was received from the DUT. At least one Door 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 door control service
GetServiceCapabilitiesResponse (door control service capabilities)
Receive and validate GetServiceCapabilities Response message
GetDoorInfoListRequest (Limit = MaxLimit, no StartReference)
Send the first part of Door Information list with specified limit and start reference
GetDoorInfoListResponse (send the first part of Door Information list with specified limit and start reference, NextStartReference)
Receive and validate Door Information list
GetDoorInfoListRequest (Limit = MaxLimit, StartReference = [current NextStartReference])
Send the second part of Door Information list with specified limit and start reference
GetDoorInfoListResponse (send the second part of Door Information list with specified limit and start reference, NextStartReference)
Receive and validate Door Information list
…
GetDoorInfoListRequest (Limit = MaxLimit, StartReference = [current NextStartReference])
Send the last part of Door Information list with specified limit and start reference
GetDoorInfoListResponse (send the last part of Door Information list with specified limit and start reference, no NextStartReference)
Receive and validate Door Information list
GetDoorInfoListRequest (Limit = 1, no StartReference)
Send the first part of Door Information list with specified limit and start reference
GetDoorInfoListResponse (send the first part of Door Information list with specified limit and start reference, NextStartReference)
Receive and validate Door Information list


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

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve DUT door control service capabilities.
4. Verify the GetServiceCapabilitiesResponse (**MaxLimit**) message from the DUT.
5. ONVIF Client will invoke GetDoorInfoListRequest message (**Limit = MaxLimit, no StartReference**) to retrieve the first part of Door Information list from the DUT.
6. Verify the GetDoorInfoListResponse message from the DUT. Verify that GetDoorInfoListResponse message does not contain number of items greater than Limit. Verify that GetDoorInfoListResponse message does not contain number of items greater than MaxLimit.
7. If the number of DoorInfo in GetDoorInfoListResponse message contains NextStartReference**,** repeat steps 5-6 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting complete door list.
8. Verify that GetDoorInfoListResponse messages at step 6 does not contain DoorInfo items with the same tokens values all over the list, i.e. there shall be no GetDoorInfoListResponse messages containing the same value for DoorInfo items token.
9. ONVIF Client will invoke GetDoorInfoListRequest message (**Limit = 1, no StartReference**) to retrieve the first part of Door Information list from the DUT.
10. Verify the GetDoorInfoListResponse message from the DUT. Verify that GetDoorInfoListResponse message does not contain number of items greater than Limit. Verify that GetDoorInfoListResponse message does not contain number of items greater than MaxLimit.
11. If the number of DoorInfo in GetDoorInfoListResponse message contains NextStartReference, repeat steps 9-10 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting complete door list.
12. Verify that GetDoorInfoListResponse messages at step 10 does not contain DoorInfo items with the same tokens values all over the list, i.e. there shall be no GetDoorInfoListResponse messages containing the same value for DoorInfo 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 GetDoorInfoListRequest message (**Limit = [limit_value], no StartReference**, where limit_value is between 1 and MaxLimit) to retrieve the first part of Door Information list from the DUT.
15. Verify the GetDoorInfoListResponse message from the DUT. Verify that GetDoorInfoListResponse message does not contain number of items greater than Limit. Verify that GetDoorInfoListResponse message does not contain number of items greater than MaxLimit.
16. If the number of DoorInfo in GetDoorInfoListResponse message contains NextStartReference, repeat steps 14-15 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting complete door list.

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

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

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

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

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

The DUT returned more DoorInfo items in GetDoorInfoListResponse message than it was specified in Limit parameter.

The DUT returned more DoorInfo items in GetDoorInfoListResponse message than MaxLimit.

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

4.2.7 GET DOOR INFO LIST – NO LIMIT

Test Label: Get Door Info List Verification without Limit.

Test Case ID: DOORCONTROL-2-1-8

ONVIF Core Specification Coverage: GetDoorInfoList (ONVIF Door Control Service Specification)

Command under test: GetDoorInfoList

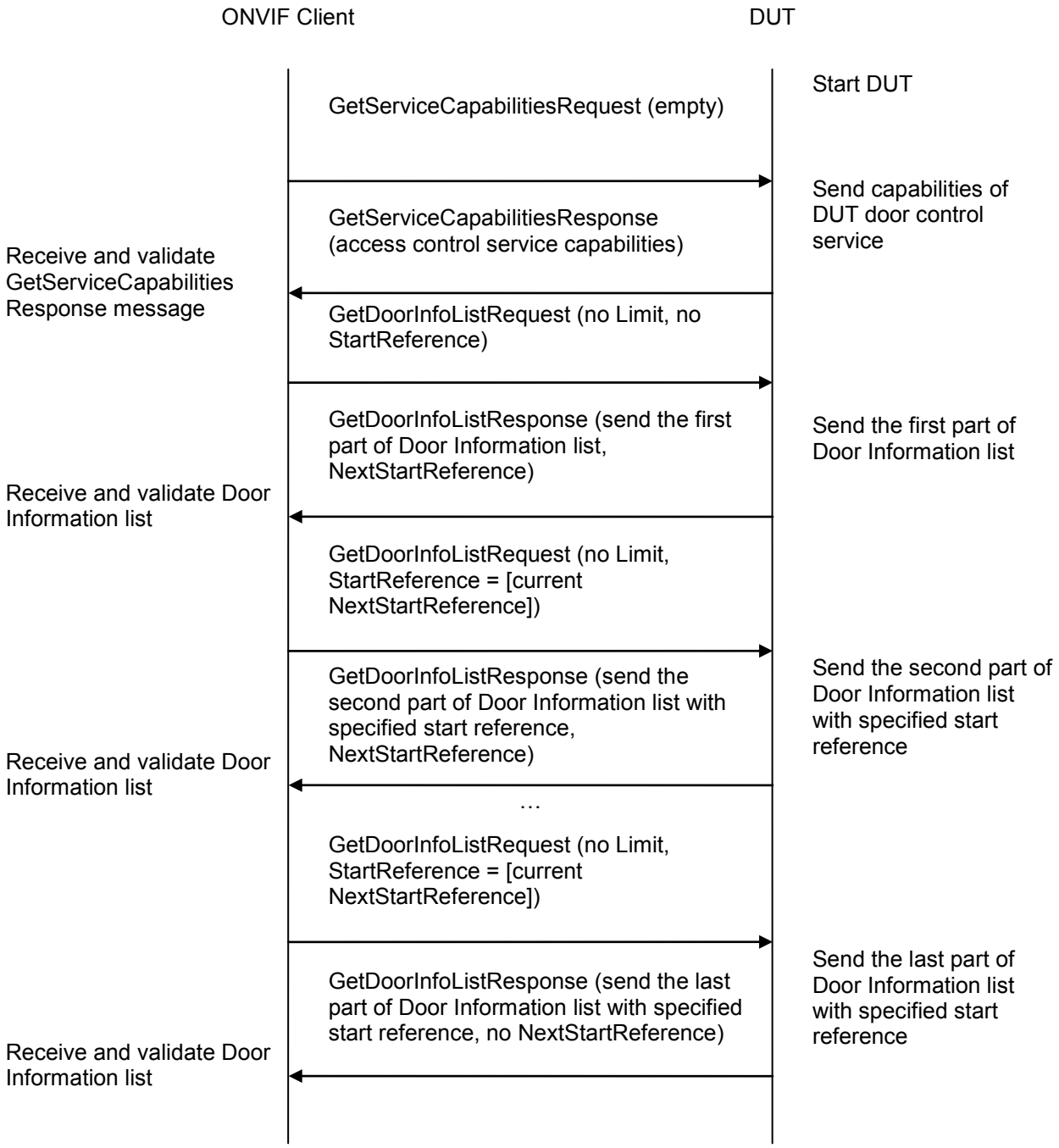
WSDL Reference: doorcontrol.wsdl

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

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

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

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

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetDoorInfoListResponse message.

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

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

The DUT returned more DoorInfo items in GetDoorInfoListResponse message than MaxLimit.

4.2.8 GET DOOR INFO – TOO MANY ITEMS

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

Test Case ID: DOORCONTROL-2-1-10

ONVIF Core Specification Coverage: GetDoorInfo (ONVIF Door Control Service Specification)

Command under test: GetDoorInfo

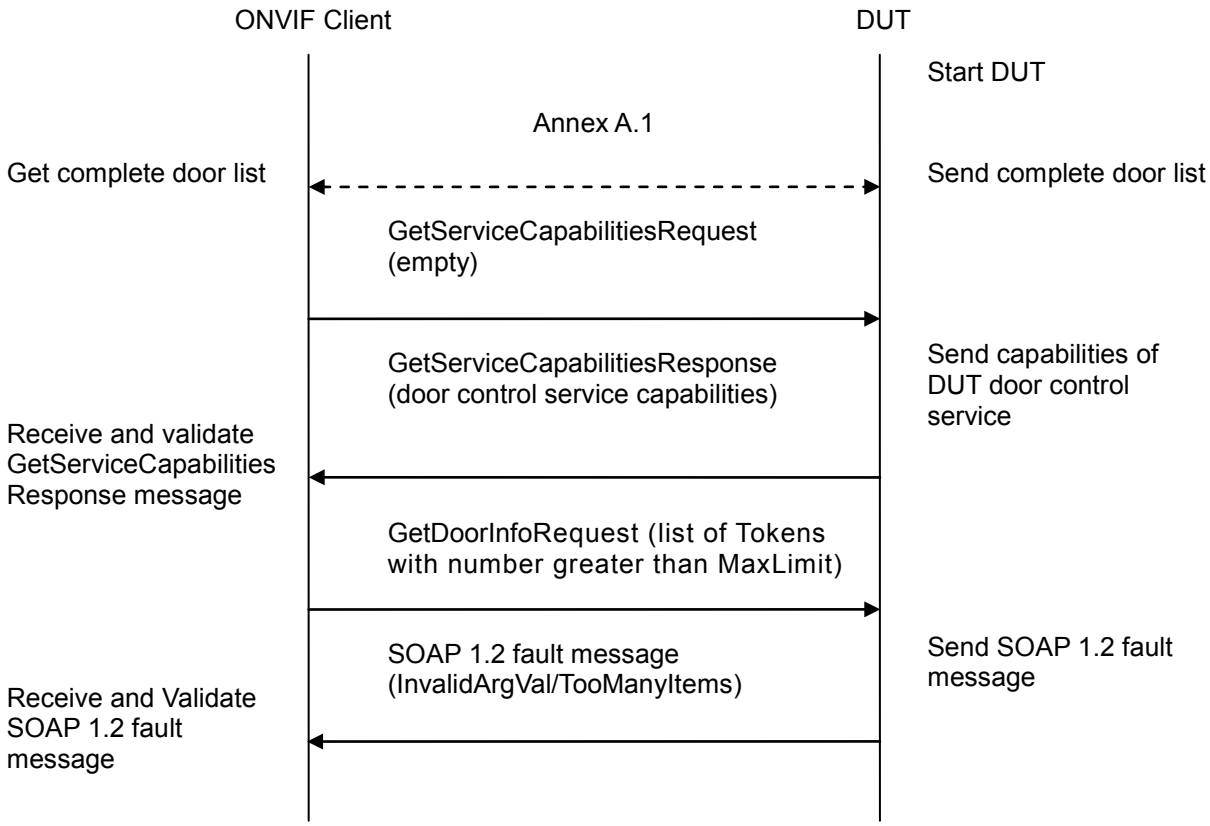
WSDL Reference: doorcontrol.wsdl

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

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors 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 doors is less than MaxLimit, skip other steps and go to the text test.
7. ONVIF Client will invoke GetDoorInfoRequest message (**Token** list with subset of DoorInfo.token values from a complete list of doors at step 3 with token number greater than MaxLimit) to retrieve a subset of Door 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 Door at step 3.

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

4.3 Door Control

4.3.1 ACCESS DOOR WITH INVALID TOKEN

Test Label: Access Door Verification with Invalid Token.

Test Case ID: DOORCONTROL-3-1-10

ONVIF Core Specification Coverage: AccessDoor (ONVIF Door Control Service Specification)

Command under test: AccessDoor

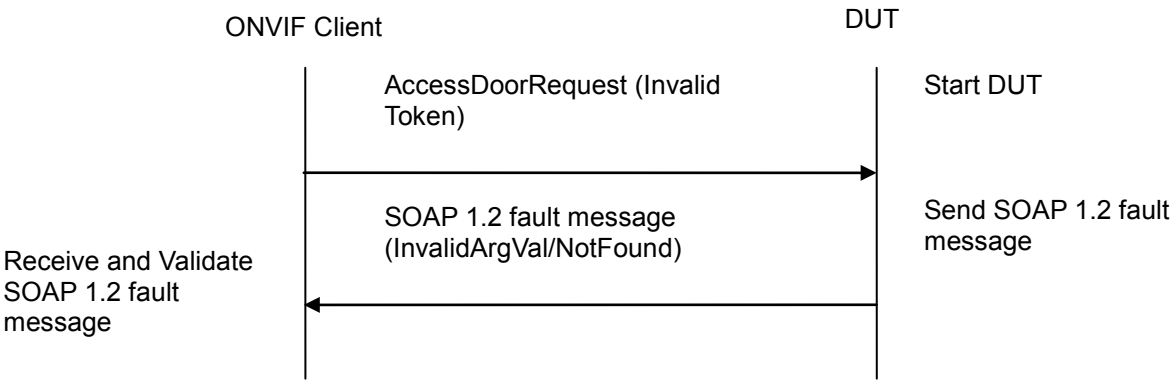
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify Access Door with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke AccessDoorRequest 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.

4.3.2 BLOCK DOOR WITH INVALID TOKEN

Test Label: Block Door Verification with Invalid Token.

Test Case ID: DOORCONTROL-3-1-11

ONVIF Core Specification Coverage: BlockDoor (ONVIF Door Control Service Specification)

Command under test: BlockDoor

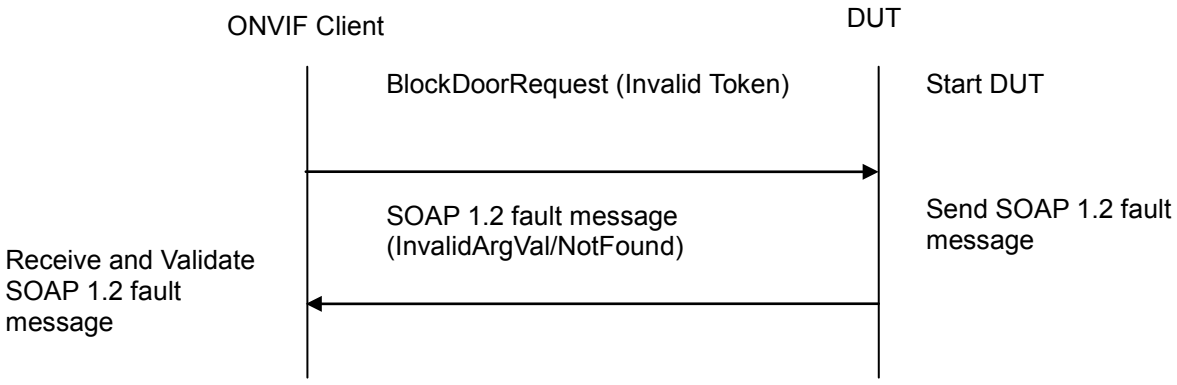
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify Block Door with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke BlockDoorRequest 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.

4.3.3 DOUBLE LOCK DOOR WITH INVALID TOKEN

Test Label: Double Lock Door Verification with Invalid Token.

Test Case ID: DOORCONTROL-3-1-12

ONVIF Core Specification Coverage: DoubleLockDoor (ONVIF Door Control Service Specification)

Command under test: DoubleLockDoor

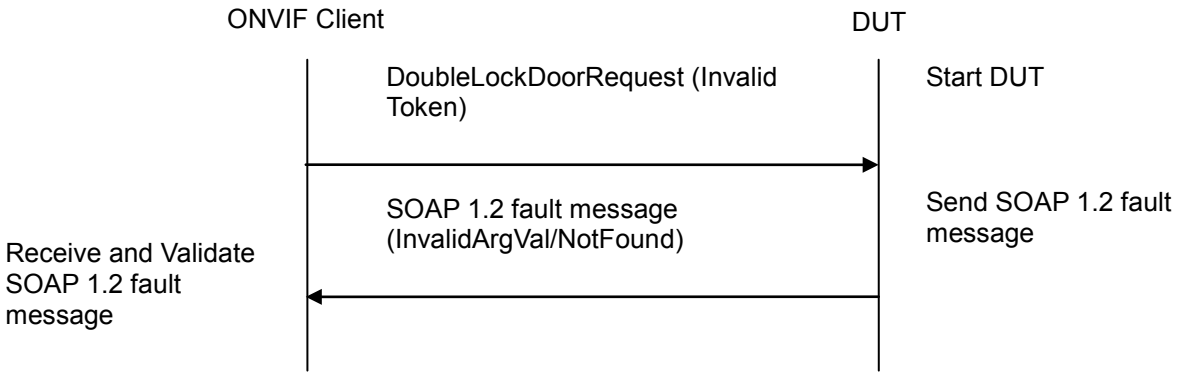
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify Double Lock Door with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke DoubleLockDoorRequest 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.

4.3.4 LOCK DOOR WITH INVALID TOKEN

Test Label: Lock Door Verification with Invalid Token.

Test Case ID: DOORCONTROL-3-1-13

ONVIF Core Specification Coverage: LockDoor (ONVIF Door Control Service Specification)

Command under test: LockDoor

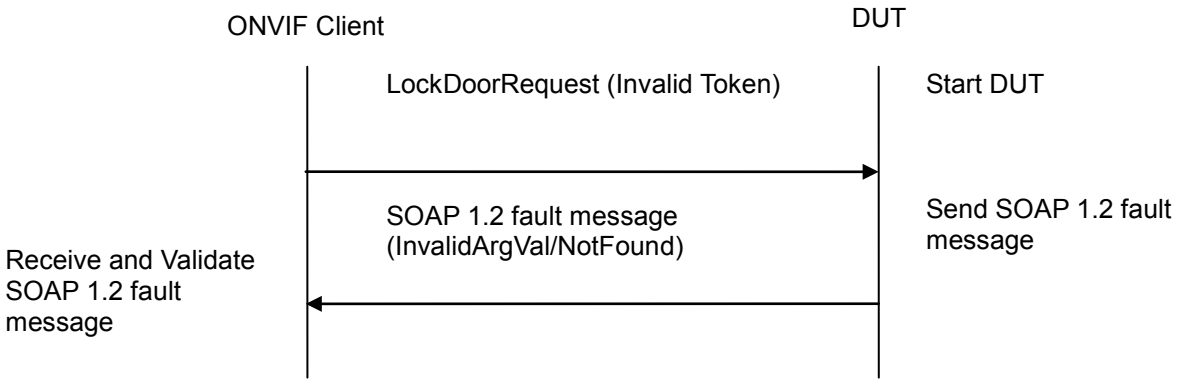
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify Lock Door with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke LockDoorRequest 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.

4.3.5 UNLOCK DOOR WITH INVALID TOKEN

Test Label: Unlock Door Verification with Invalid Token.

Test Case ID: DOORCONTROL-3-1-14

ONVIF Core Specification Coverage: UnlockDoor (ONVIF Door Control Service Specification)

Command under test: UnlockDoor

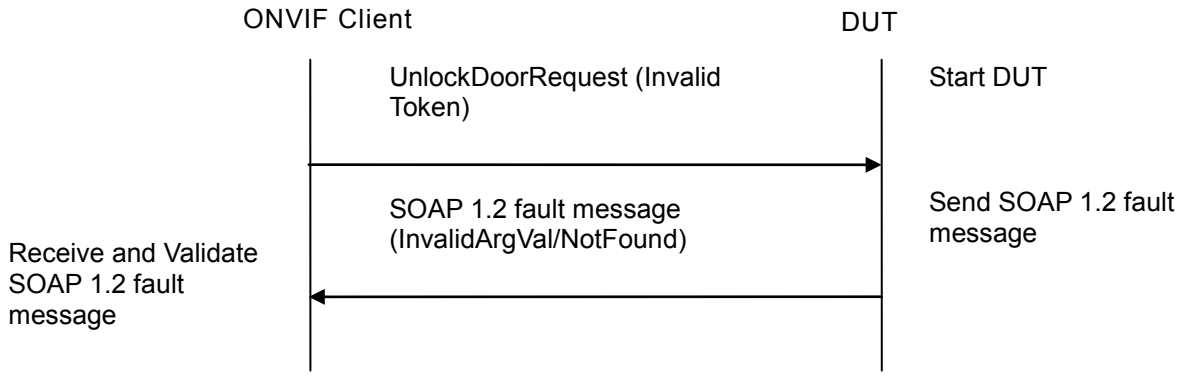
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify Unlock Door with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke UnlockDoorRequest 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.

4.3.6 LOCK DOWN DOOR WITH INVALID TOKEN

Test Label: Lock Down Door Verification with Invalid Token.

Test Case ID: DOORCONTROL-3-1-15

ONVIF Core Specification Coverage: LockDownDoor (ONVIF Door Control Service Specification)

Command under test: LockDownDoor

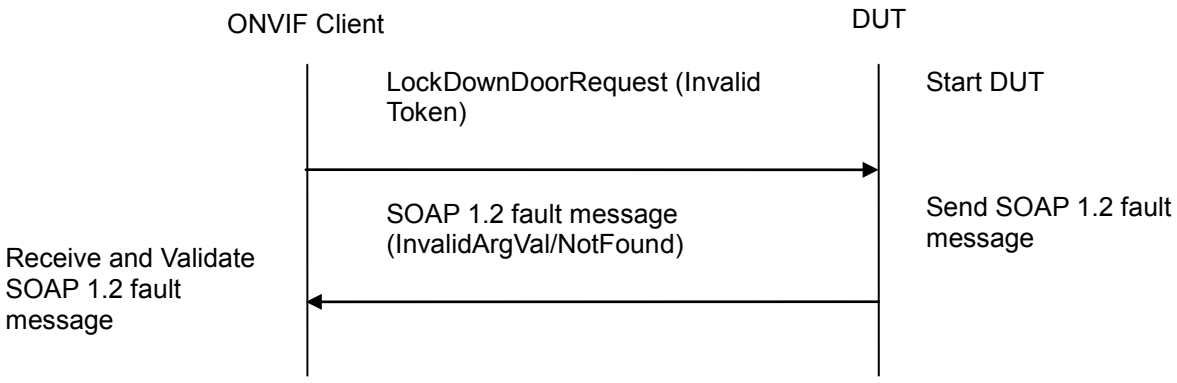
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify Lock Down Door with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke LockDownDoorRequest 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.

4.3.7 LOCK DOWN RELEASE DOOR WITH INVALID TOKEN

Test Label: Lock Down Release Door Verification with Invalid Token.

Test Case ID: DOORCONTROL-3-1-16

ONVIF Core Specification Coverage: LockDownReleaseDoor (ONVIF Door Control Service Specification)

Command under test: LockDownReleaseDoor

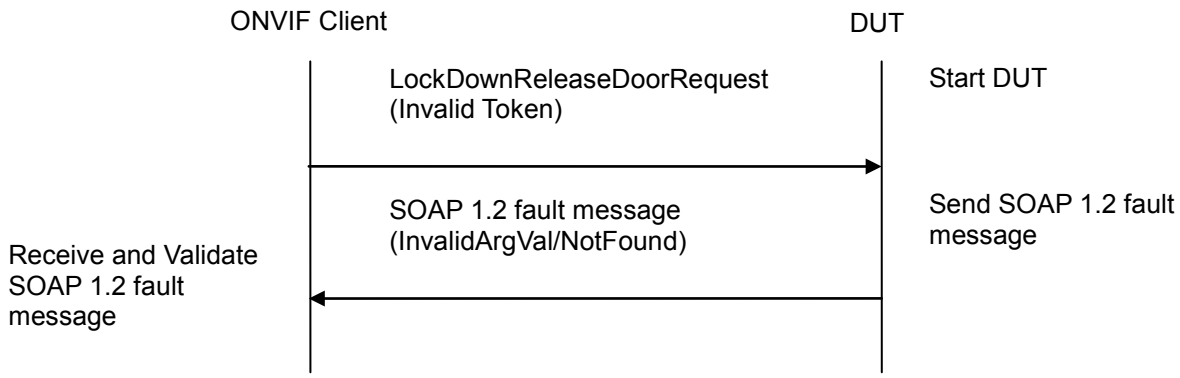
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify Lock Down Release Door with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

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

3. ONVIF Client will invoke LockDownReleaseDoorRequest 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.

4.3.8 LOCK OPEN DOOR WITH INVALID TOKEN

Test Label: Lock Open Door Verification with Invalid Token.

Test Case ID: DOORCONTROL-3-1-17

ONVIF Core Specification Coverage: LockOpenDoor (ONVIF Door Control Service Specification)

Command under test: LockOpenDoor

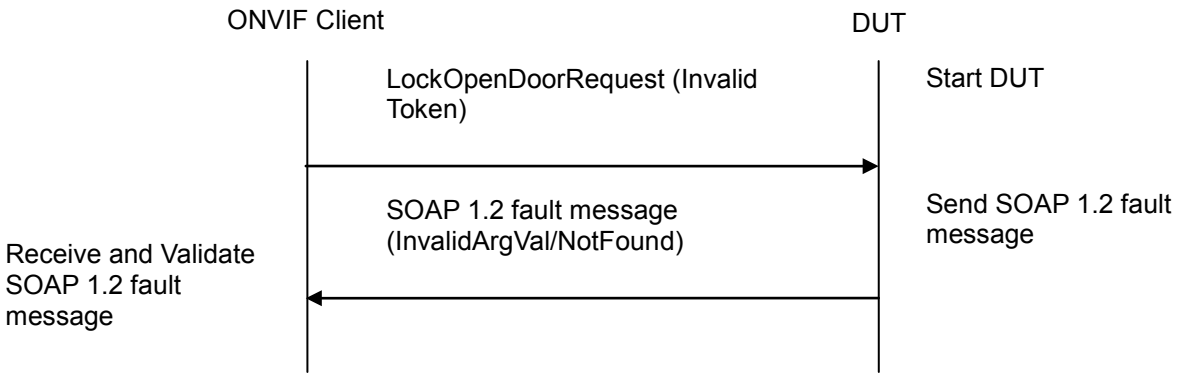
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify Lock Open Door with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

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

3. ONVIF Client will invoke LockOpenDoorRequest 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.

4.3.9 LOCK OPEN RELEASE DOOR WITH INVALID TOKEN

Test Label: Lock Open Release Door Verification with Invalid Token.

Test Case ID: DOORCONTROL-3-1-18

ONVIF Core Specification Coverage: LockOpenReleaseDoor (ONVIF Door Control Service Specification)

Command under test: LockOpenReleaseDoor

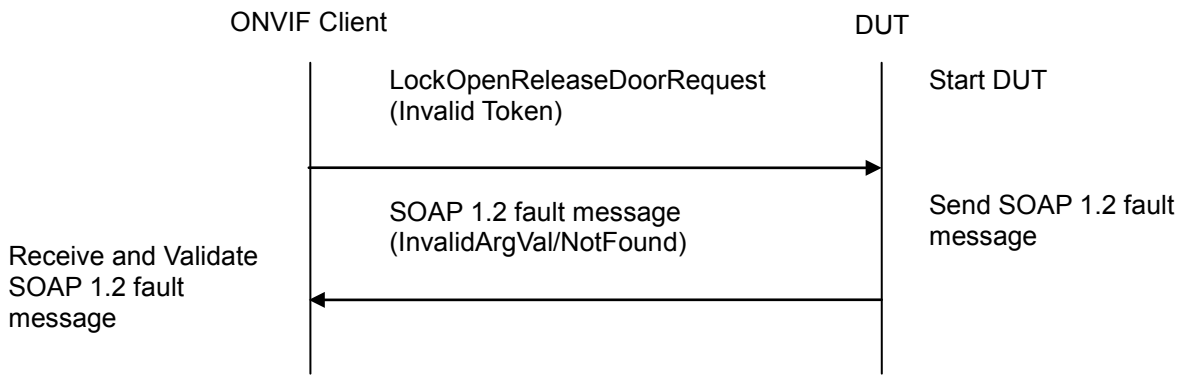
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify Lock Open Release Door with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.
3. ONVIF Client will invoke LockOpenReleaseDoorRequest 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.

4.3.10 ACCESS DOOR – COMMAND NOT SUPPORTED

Test Label: Access Door Verification – Command not Supported.

Test Case ID: DOORCONTROL-3-1-19

ONVIF Core Specification Coverage: AccessDoor (ONVIF Door Control Service Specification)

Command under test: AccessDoor

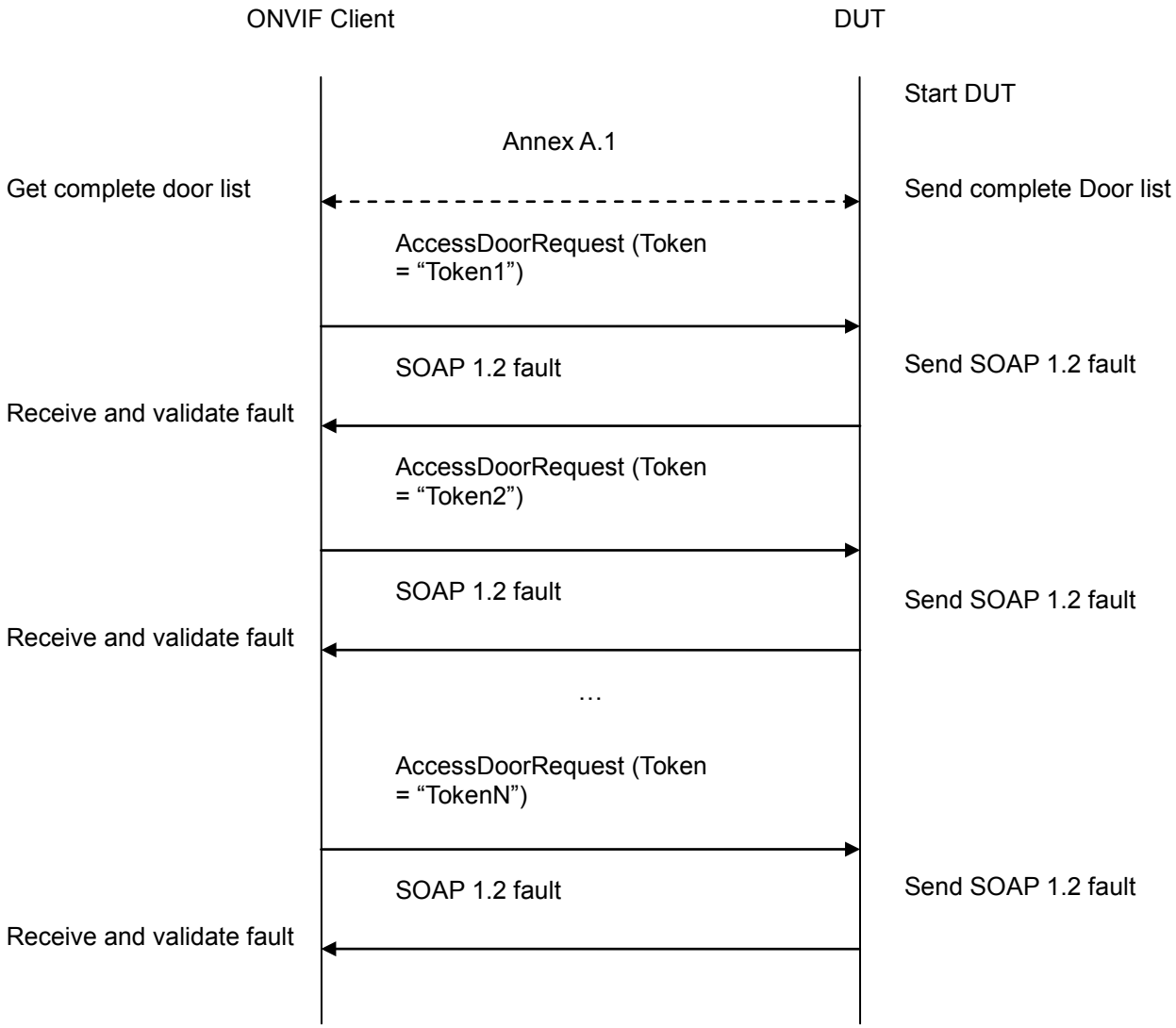
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that fault is returned for Access Door in case Door does not support it.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. If Door with **Token1** (Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) has **DoorInfo.Capabilities.Access** equal to **true**, then skip steps 5-6 and go to the step 7.
5. ONVIF Client will invoke AccessDoorRequest message (**Token = “Token1”**, where Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) to try changing door state.
6. The DUT will generate SOAP 1.2 fault message.
7. Repeat steps 4-6 for all other tokens from the complete list of doors 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 Door at step 3.

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

4.3.11 BLOCK DOOR – COMMAND NOT SUPPORTED

Test Label: Block Door Verification – Command not Supported.

Test Case ID: DOORCONTROL-3-1-20

ONVIF Core Specification Coverage: BlockDoor (ONVIF Door Control Service Specification)

Command under test: BlockDoor

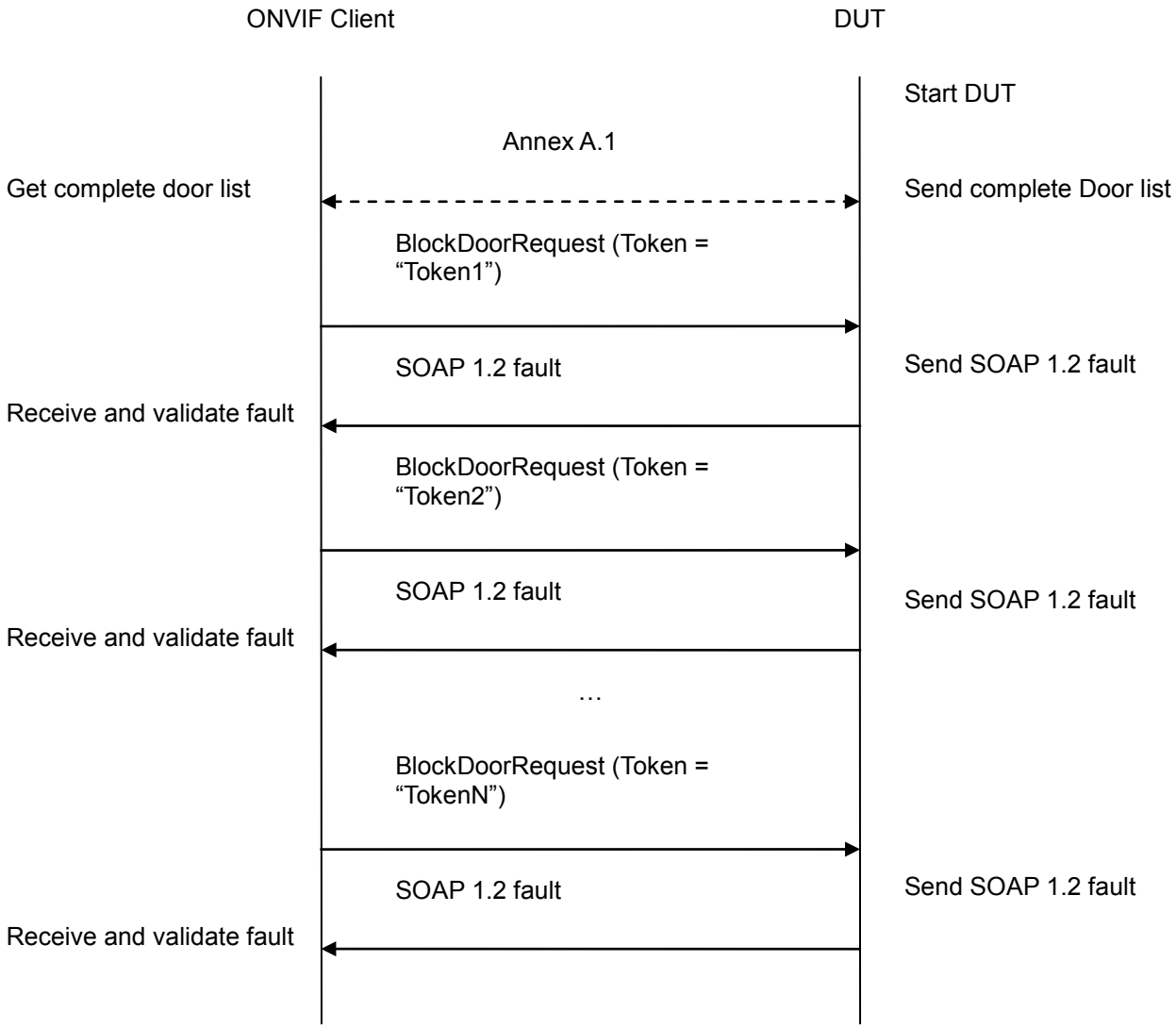
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that fault is returned for Block Door in case Door does not support it.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. If Door with **Token1** (Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) has **DoorInfo.Capabilities.Block** equal to **true**, then skip steps 5-6 and go to the step 7.
5. ONVIF Client will invoke BlockDoorRequest message (**Token = "Token1"**, where Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) to try changing door state.
6. The DUT will generate SOAP 1.2 fault message.
7. Repeat steps 5-7 for all other tokens from the complete list of doors 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 Door at step 3.

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

4.3.12 DOUBLE LOCK DOOR – COMMAND NOT SUPPORTED

Test Label: Double Lock Door Verification – Command not Supported.

Test Case ID: DOORCONTROL-3-1-21

ONVIF Core Specification Coverage: DoubleLockDoor (ONVIF Door Control Service Specification)

Command under test: DoubleLockDoor

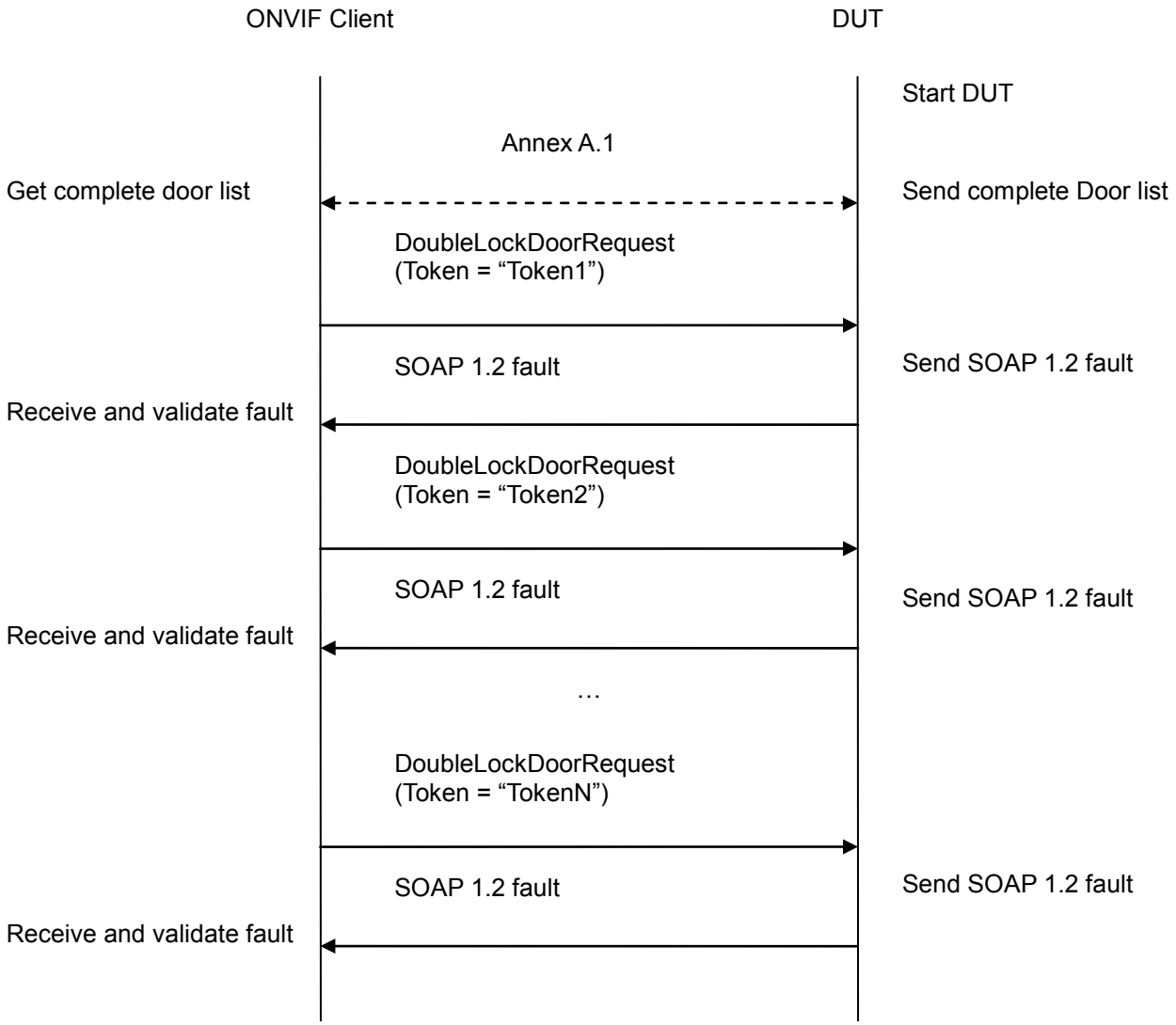
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that fault is returned for Double Lock Door in case Door does not support it.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. If Door with **Token1** (Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) has **DoorInfo.Capabilities.DoubleLock** equal to **true**, then skip steps 5-6 and go to the step 7.
5. ONVIF Client will invoke DoubleLockDoorRequest message (**Token = “Token1”**, where Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) to try changing door state.
6. The DUT will generate SOAP 1.2 fault message.
7. Repeat steps 5-7 for all other tokens from complete list of doors 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 Door at step 3.

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

4.3.13 LOCK DOOR – COMMAND NOT SUPPORTED

Test Label: Lock Door Verification – Command not Supported.

Test Case ID: DOORCONTROL-3-1-22

ONVIF Core Specification Coverage: LockDoor (ONVIF Door Control Service Specification)

Command under test: LockDoor

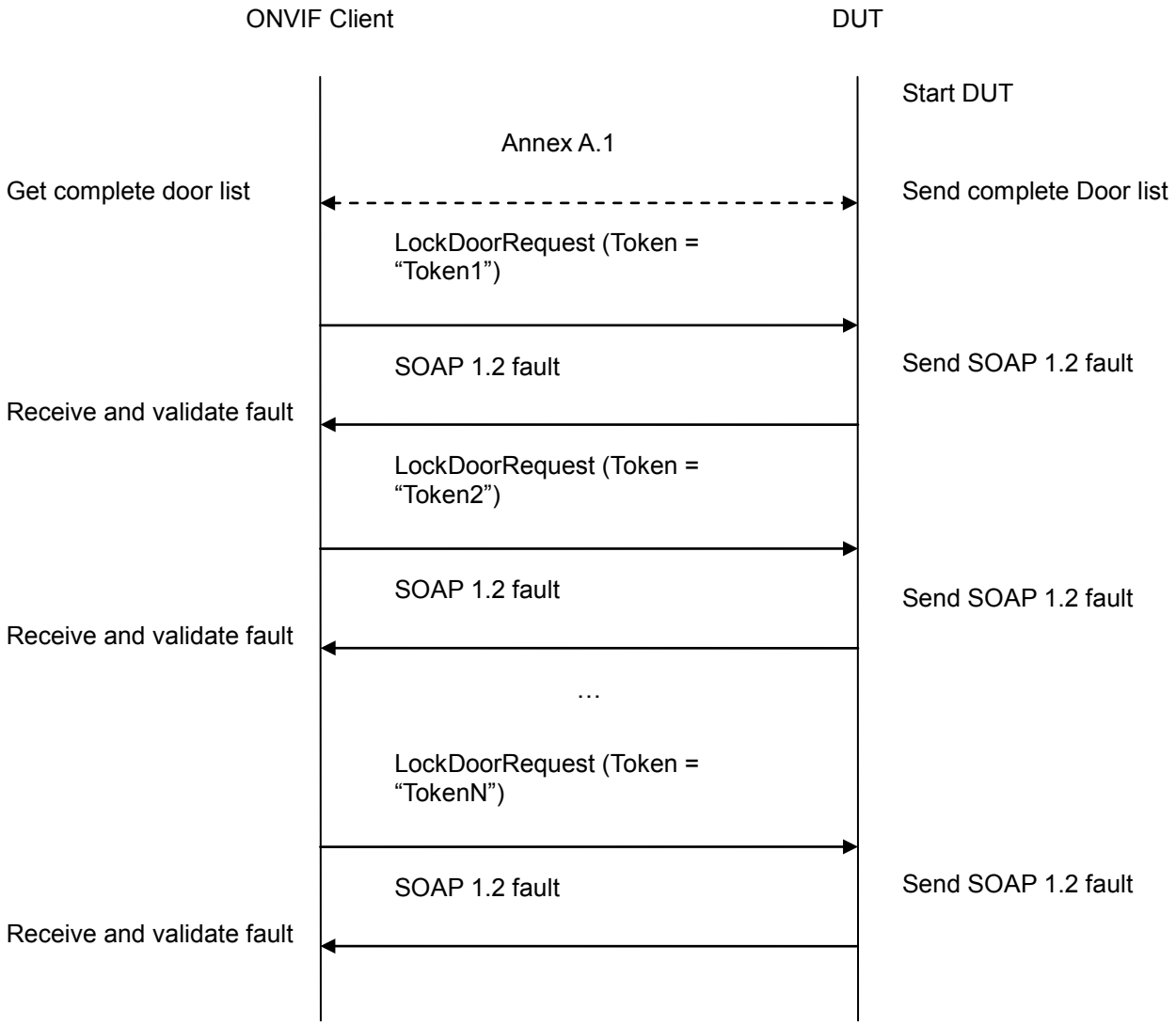
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that fault is returned for Lock Door in case Door does not support it.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. If Door with **Token1** (Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) has **DoorInfo.Capabilities.Lock** equal to **true**, then skip steps 5-6 and go to the step 7.
5. ONVIF Client will invoke LockDoorRequest message (**Token = "Token1"**, where Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) to try changing door state.
6. The DUT will generate SOAP 1.2 fault message.
7. Repeat steps 5-7 for all other tokens from the complete list of doors 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 Door at step 3.

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

4.3.14 UNLOCK DOOR – COMMAND NOT SUPPORTED

Test Label: Unlock Door Verification – Command not Supported.

Test Case ID: DOORCONTROL-3-1-23

ONVIF Core Specification Coverage: UnlockDoor (ONVIF Door Control Service Specification)

Command under test: UnlockDoor

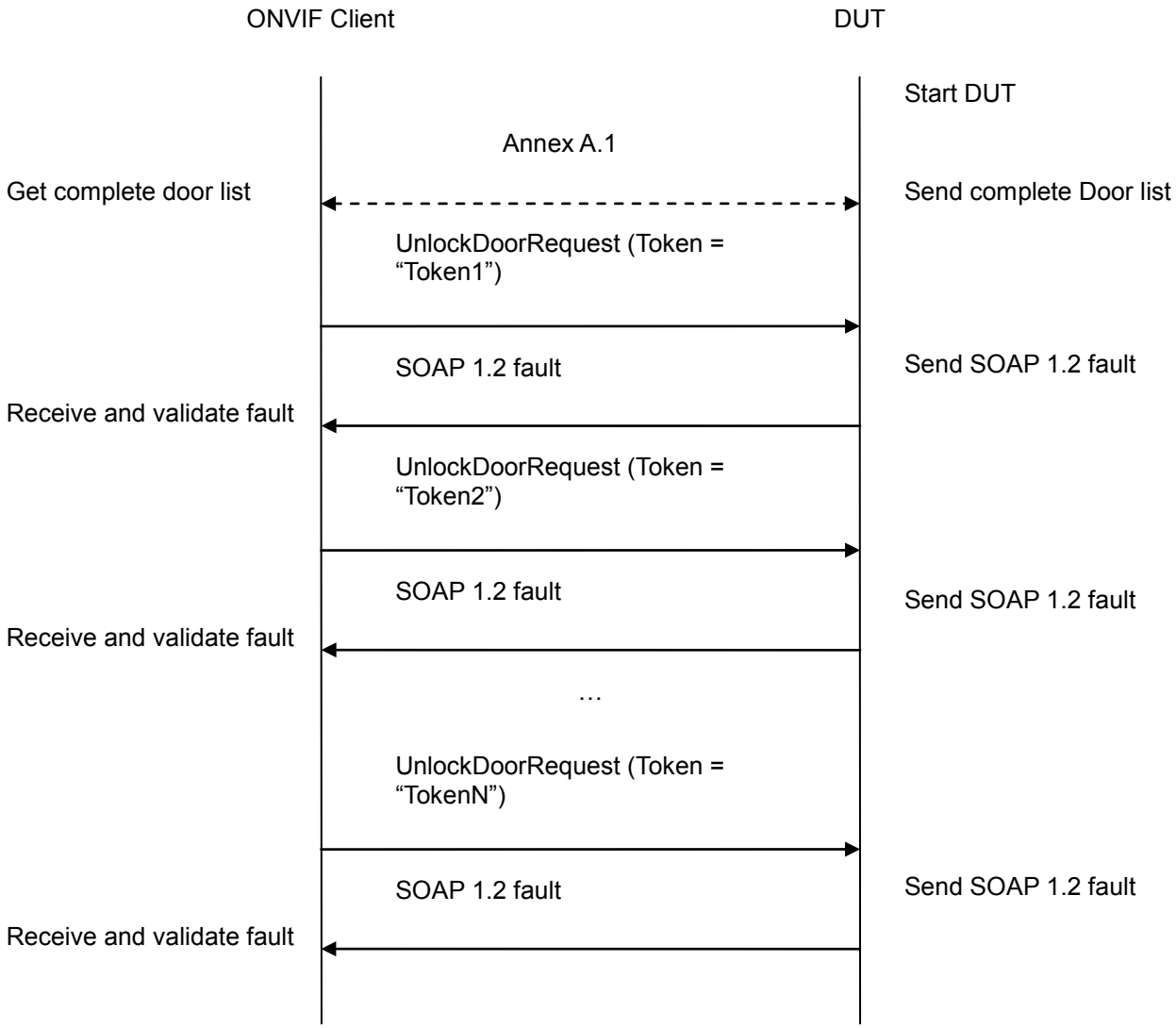
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that fault is returned for Unlock Door in case Door does not support it.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. If Door with **Token1** (Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) has **DoorInfo.Capabilities.Unlock** equal to **true**, then skip steps 5-6 and go to the step 7.
5. ONVIF Client will invoke UnlockDoorRequest message (**Token = "Token1"**, where Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) to try changing door state.
6. The DUT will generate SOAP 1.2 fault message.
7. Repeat steps 5-7 for all other tokens from the complete list of doors 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 Door at step 3.

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

4.3.15 LOCK DOWN DOOR – COMMAND NOT SUPPORTED

Test Label: Lock Down Door Verification – Command not Supported.

Test Case ID: DOORCONTROL-3-1-24

ONVIF Core Specification Coverage: LockDownDoor (ONVIF Door Control Service Specification)

Command under test: LockDownDoor

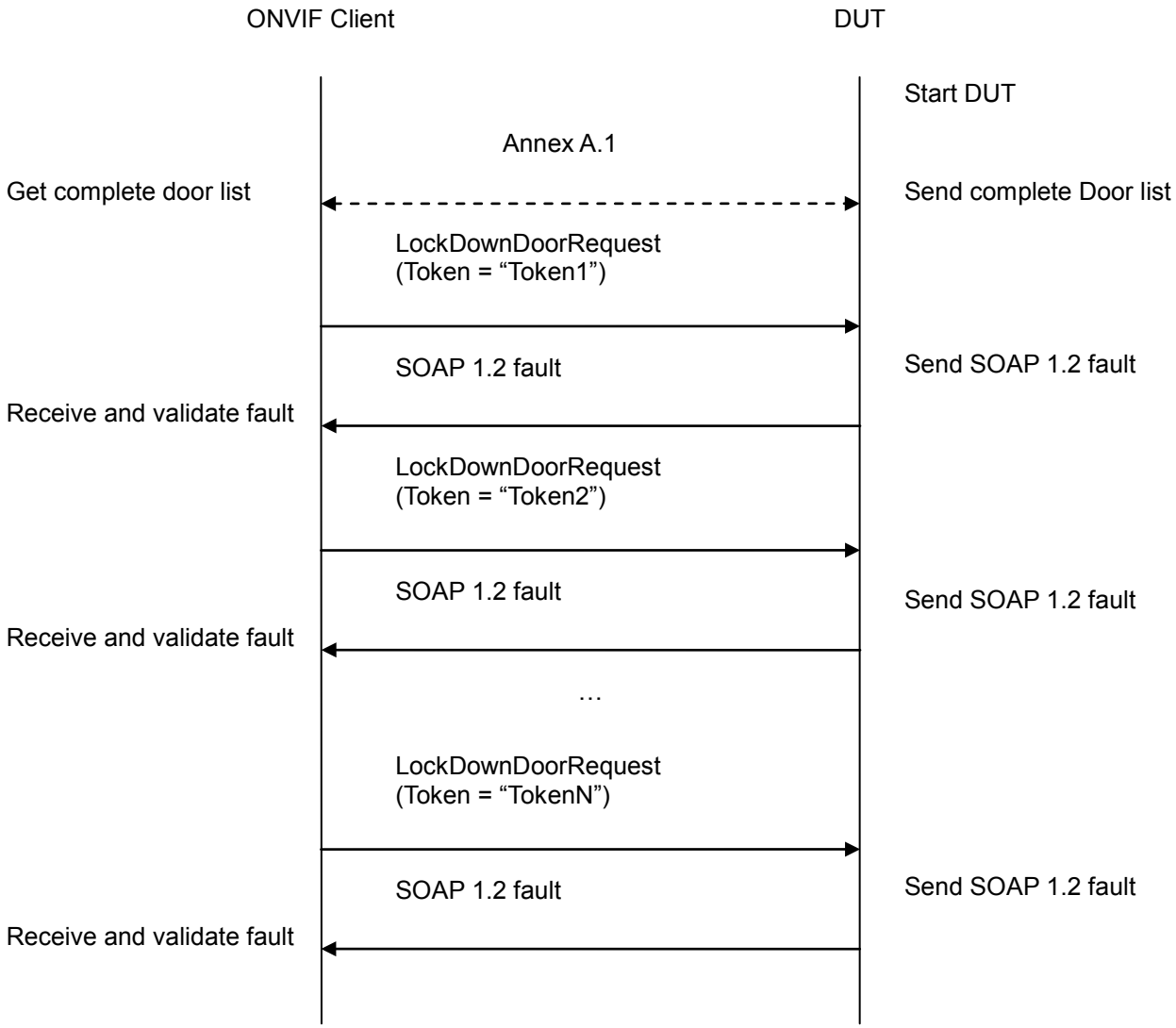
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that fault is returned for Lock Down Door in case Door does not support it.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. If Door with **Token1** (Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) has **DoorInfo.Capabilities.LockDown** equal to **true**, then skip steps 5-6 and go to the step 7.
5. ONVIF Client will invoke LockDownDoorRequest message (**Token = “Token1”**, where Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) to try changing door state.
6. The DUT will generate SOAP 1.2 fault message.
7. Repeat steps 5-7 for all other tokens from the complete list of doors 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 Door at step 3.

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

4.3.16 LOCK DOWN RELEASE DOOR – COMMAND NOT SUPPORTED

Test Label: Lock Down Release Door Verification – Command not Supported.

Test Case ID: DOORCONTROL-3-1-25

ONVIF Core Specification Coverage: LockDownReleaseDoor (ONVIF Door Control Service Specification)

Command under test: LockDownReleaseDoor

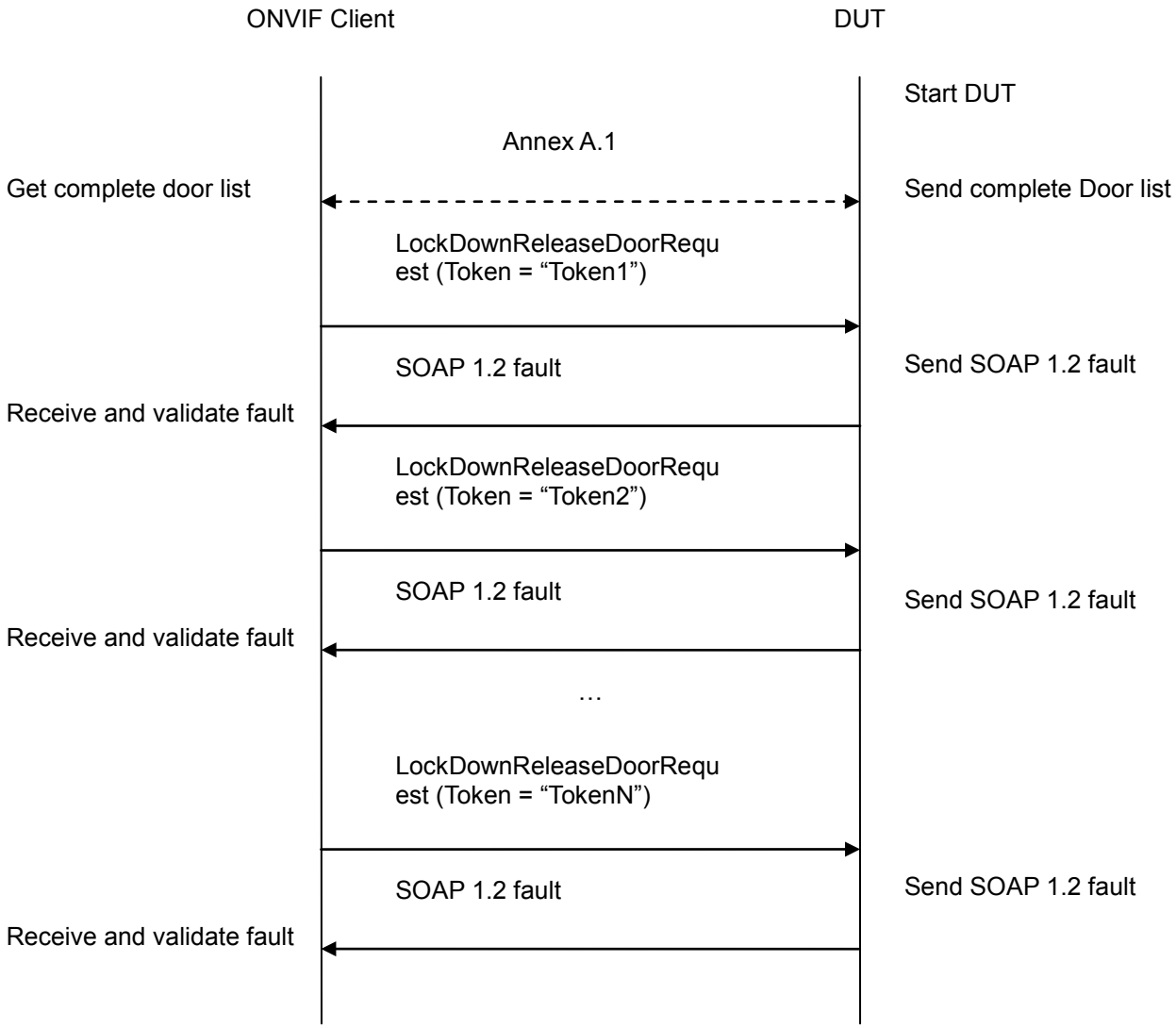
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that fault is returned for Lock Down Release Door in case Door does not support it.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. If Door with **Token1** (Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) has **DoorInfo.Capabilities.LockDown** equal to **true**, then skip steps 5-6 and go to the step 7.
5. ONVIF Client will invoke LockDownReleaseDoorRequest message (**Token = “Token1”**, where Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) to try changing door state.
6. The DUT will generate SOAP 1.2 fault message.
7. Repeat steps 5-7 for all other tokens from the complete list of doors 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 Door at step 3.

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

4.3.17 LOCK OPEN DOOR – COMMAND NOT SUPPORTED

Test Label: Lock Open Door Verification – Command not Supported.

Test Case ID: DOORCONTROL-3-1-26

ONVIF Core Specification Coverage: LockOpenDoor (ONVIF Door Control Service Specification)

Command under test: LockOpenDoor

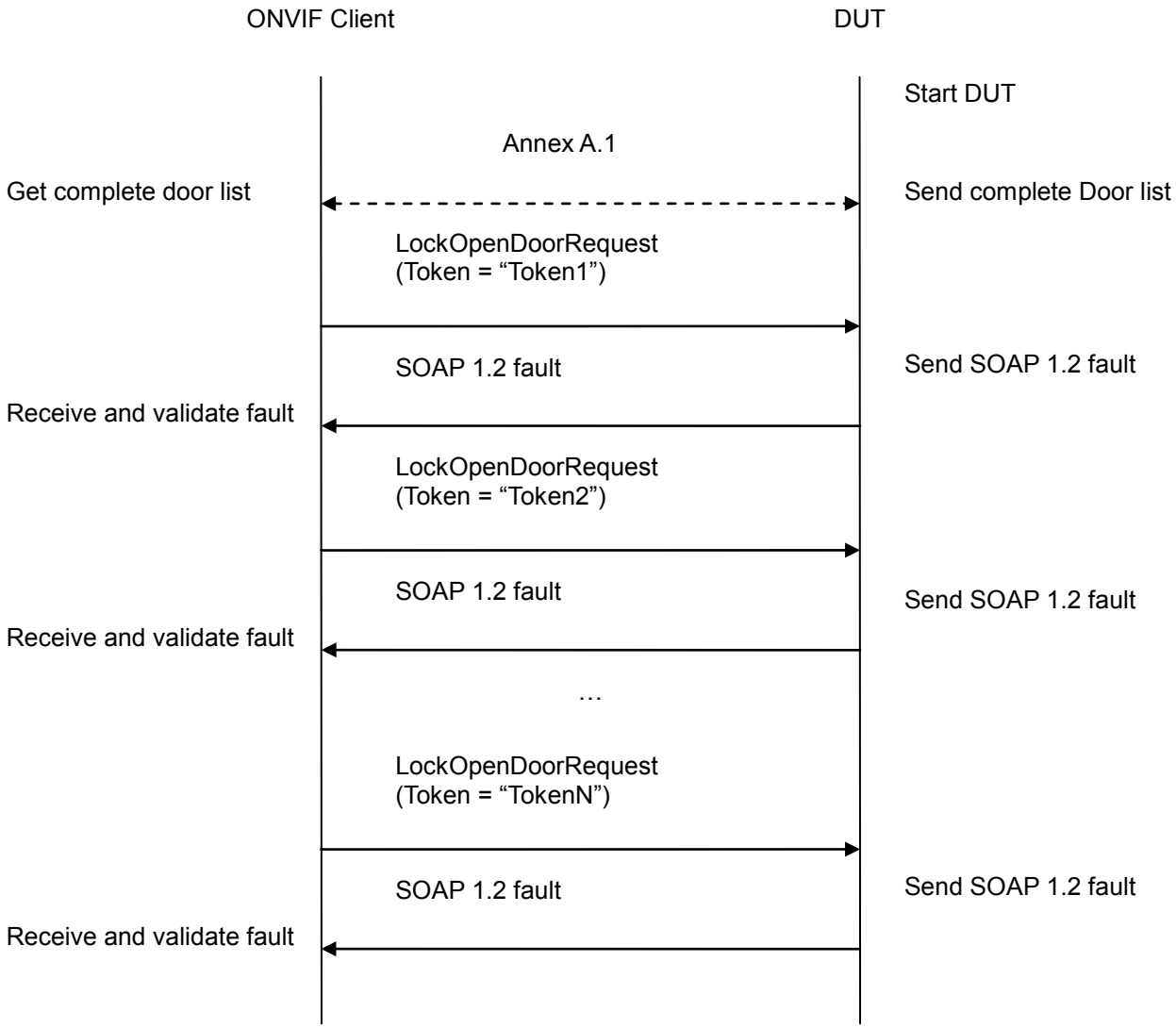
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that fault is returned for Lock Open Door in case Door does not support it.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. If Door with **Token1** (Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) has **DoorInfo.Capabilities.LockOpen** equal to **true**, then skip steps 5-6 and go to the step 7.
5. ONVIF Client will invoke LockOpenDoorRequest message (**Token = “Token1”**, where Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) to try changing door state.
6. The DUT will generate SOAP 1.2 fault message.
7. Repeat steps 5-7 for all other tokens from the complete list of doors 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 Door at step 3.

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

4.3.18 LOCK OPEN RELEASE DOOR – COMMAND NOT SUPPORTED

Test Label: Lock Open Release Door Verification – Command not Supported.

Test Case ID: DOORCONTROL-3-1-27

ONVIF Core Specification Coverage: LockOpenReleaseDoor (ONVIF Door Control Service Specification)

Command under test: LockOpenReleaseDoor

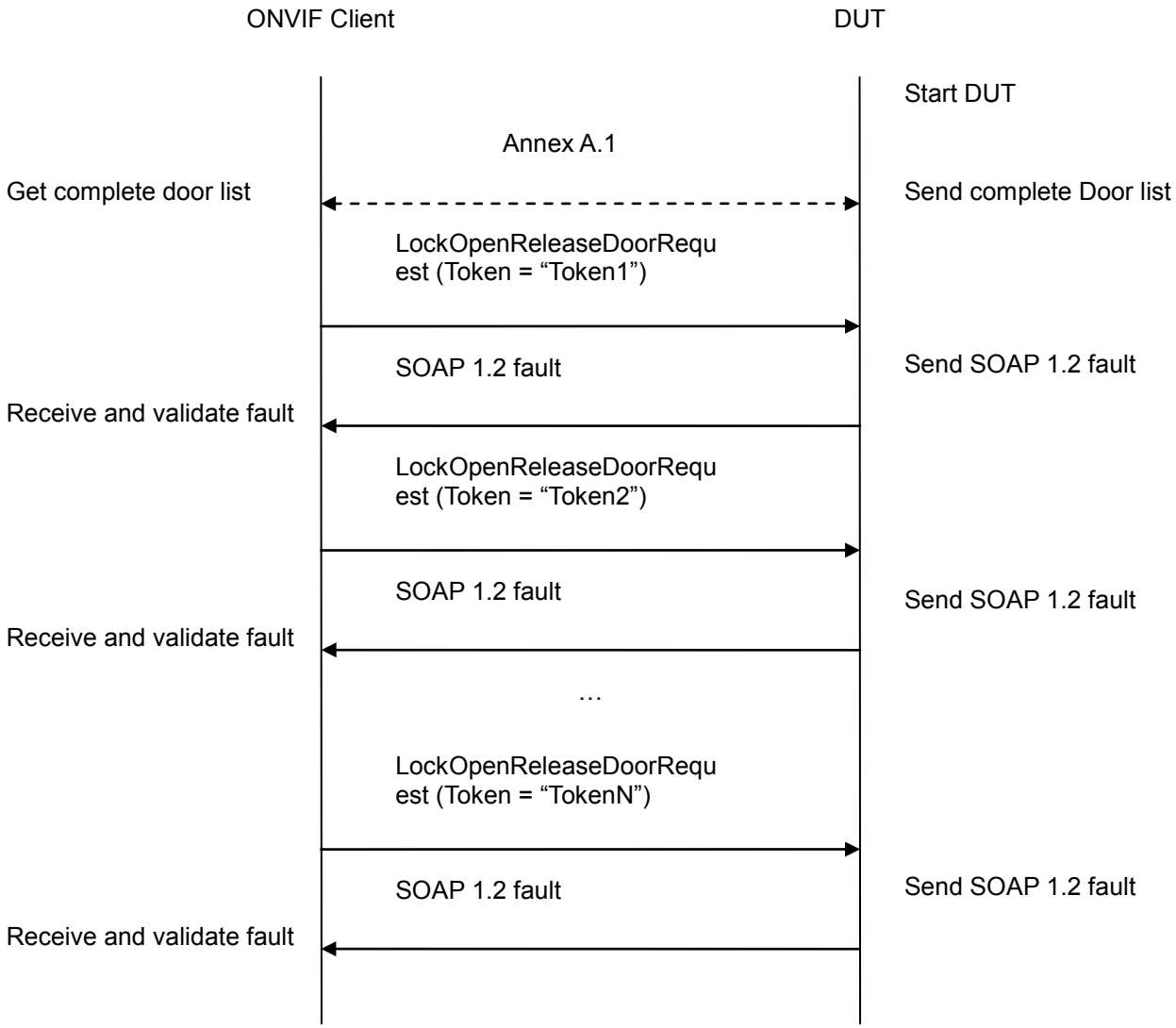
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that fault is returned for Lock Open Release Door in case Door does not support it.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. If Door with **Token1** (Token1 is the first **DoorInfo.token** from the complete list of doors at step 3) has **DoorInfo.Capabilities.LockOpen** equal to **true**, then skip steps 5-6 and go to the step 7.
5. ONVIF Client will invoke LockOpenReleaseDoorRequest message (**Token = “Token1”**, where Token1 is the first **DoorInfo.token** from complete list of doors at step 3) to try changing door state.
6. The DUT will generate SOAP 1.2 fault message.
7. Repeat steps 5-7 for all other tokens from the complete list of doors 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 Door at step 3.

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

4.3.19 ACCESS DOOR

Test Label: Access Door Verification.

Test Case ID: DOORCONTROL-3-1-28

ONVIF Core Specification Coverage: AccessDoor (ONVIF Door Control Service Specification), tns1:Door/State/DoorMode (ONVIF Door Control Service Specification)

Command under test: AccessDoor

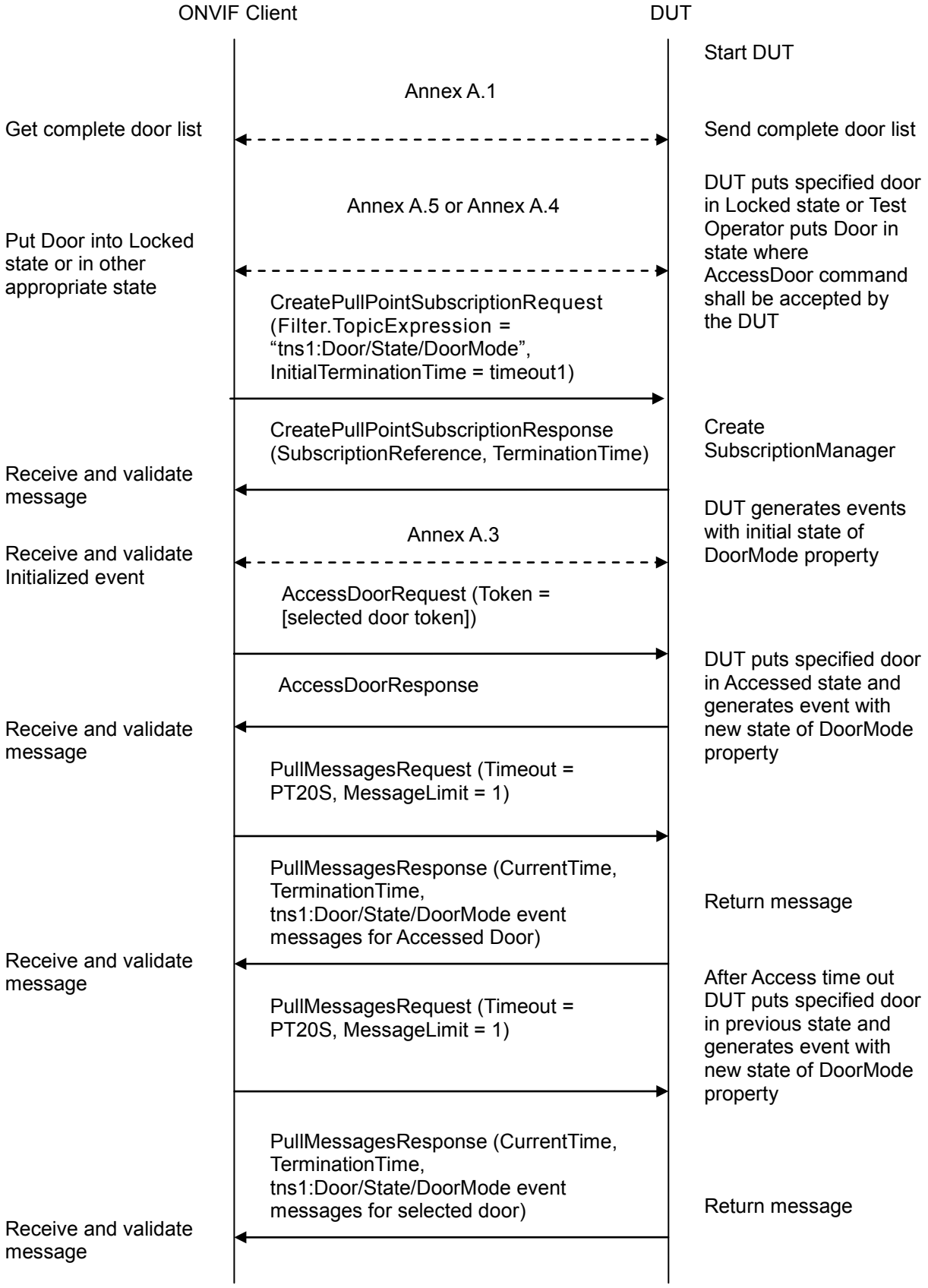
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that Access Door is accepted. To verify Door state change after Access Door command. To verify tns1:Door/State/DoorMode property events.

Pre-Requisite: Door Control Service was received from the DUT. At least one Door with Access capability is configured and added to the DUT. The doors are closed and there are no schedules.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete door list
Send complete door list
DUT puts specified door in Locked state or Test Operator puts Door in state where AccessDoor command shall be accepted by the DUT
Annex A.5 or Annex A.4
Put Door into Locked state or in other appropriate state
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/DoorMode”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
DUT generates events with initial state of DoorMode property
Annex A.3
Receive and validate Initialized event
AccessDoorRequest (Token = [selected door token])
DUT puts specified door in Accessed state and generates event with new state of DoorMode property
AccessDoorResponse
Receive and validate message
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorMode event messages for Accessed Door)
Return message
Receive and validate message
After Access time out DUT puts specified door in previous state and generates event with new state of DoorMode property
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorMode event messages for selected door)
Return message
Receive and validate message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. ONVIF Client selects one door from the complete list of doors at step 3 with **DoorInfo.Capabilities.Access** equal to **true** and **DoorInfo.Capabilities.Lock** equal to **true**. If there is no Door with **DoorInfo.Capabilities.Access** equal to **true**, fail the test and skip other steps. If there is no Door with **DoorInfo.Capabilities.Access** equal to true and **DoorInfo.Capabilities.Lock** equal to **true,** then ONVIF Client executes Annex A.4 to start user interaction process, and then goes to step 6**.**
5. ONVIF Client will execute Annex A.5 to switch the selected door into Locked state.
6. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
8. Execute Annex A.3 to catch Initialized event for selected door. State of the Door received in Annex A.3 will be assumed as initial State of this Door.
9. ONVIF Client will invoke AccessDoorRequest message (**Token = [selected Door token]**) to change door state.
10. Verify the AccessDoorResponse message from the DUT.
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 NotificationMessages. Repeat step 11 until Notification messages with PropertyOperation = "Changed" contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to selected Door Token with State = “Accessed” is received (Operation Delay option from ONVIF Device Test Tool will be used).
13. Check that all Notification messages with PropertyOperation = "Changed" contain Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token.
14. Check that all Notification messages with PropertyOperation = "Changed" have TopicExpression equal to tns1:Door/State/DoorMode.
15. Check that received Notification message with PropertyOperation = "Changed" contains Data.SimpleItem item with Name = «State» and Value equal to “Accessed”. Check that other Notification messages contain Data.SimpleItem item with Name = «State» and Value with tdc:DoorMode type.
16. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).
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 NotificationMessages. Repeat step 17 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
19. Check that all Notification messages with PropertyOperation = "Changed" contain Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token.
20. Check that all Notification messages with PropertyOperation = "Changed" have TopicExpression equal to tns1:Door/State/DoorMode.
21. Check that the last Notification message with PropertyOperation = "Changed" contains Data.SimpleItem item with Name = «State» and Value equal to initial Door state from step 8. Check that other Notification messages contain Data.SimpleItem item with Name = «State» and Value with tdc:DoorMode type.
22. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid AccessDoorResponse message.

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not return at least one Door at step 3.

The DUT sent an invalid Notification messages with PropertyOperation = "Changed" (correct value for UTC time, TopicExpression and wsnt:Message, Topic = “tns1:Door/State/DoorMode”, contains Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token, contains Data.SimpleItem item with Name = «State» and Value with type equal to tdc:DoorMode) at steps 12 and 18.

The DUT did not send Notification message with PropertyOperation = "Changed" with State value “Accessed” at step 12.

The DUT sent the last Notification message with PropertyOperation = "Changed" at step 18 with State value other than initial State at step 8.

Note: If the DUT did not return any door for step 3, skip steps 4-22, fail the test and go to the next test.

Note: If the DUT sends SOAP fault to AccessDoorRequest message then the ONVIF Client deletes Subscription Manager, starts user interaction process (see Annex A.4), and then runs the test from step 6.

Note: If in the Annex A.5 the DUT sends SOAP fault to request to change door mode, then the ONVIF Client starts user interaction process (see Annex A.4), and then runs the test from step 6.

Note: The Subscription Manager has to be deleted at the end of the test either by calling

unsubscribe or through a timeout.

Note: 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.

4.3.20 BLOCK DOOR

Test Label: Block Door Verification.

Test Case ID: DOORCONTROL-3-1-29

ONVIF Core Specification Coverage: BlockDoor (ONVIF Door Control Service Specification), tns1:Door/State/DoorMode (ONVIF Door Control Service Specification)

Command under test: BlockDoor

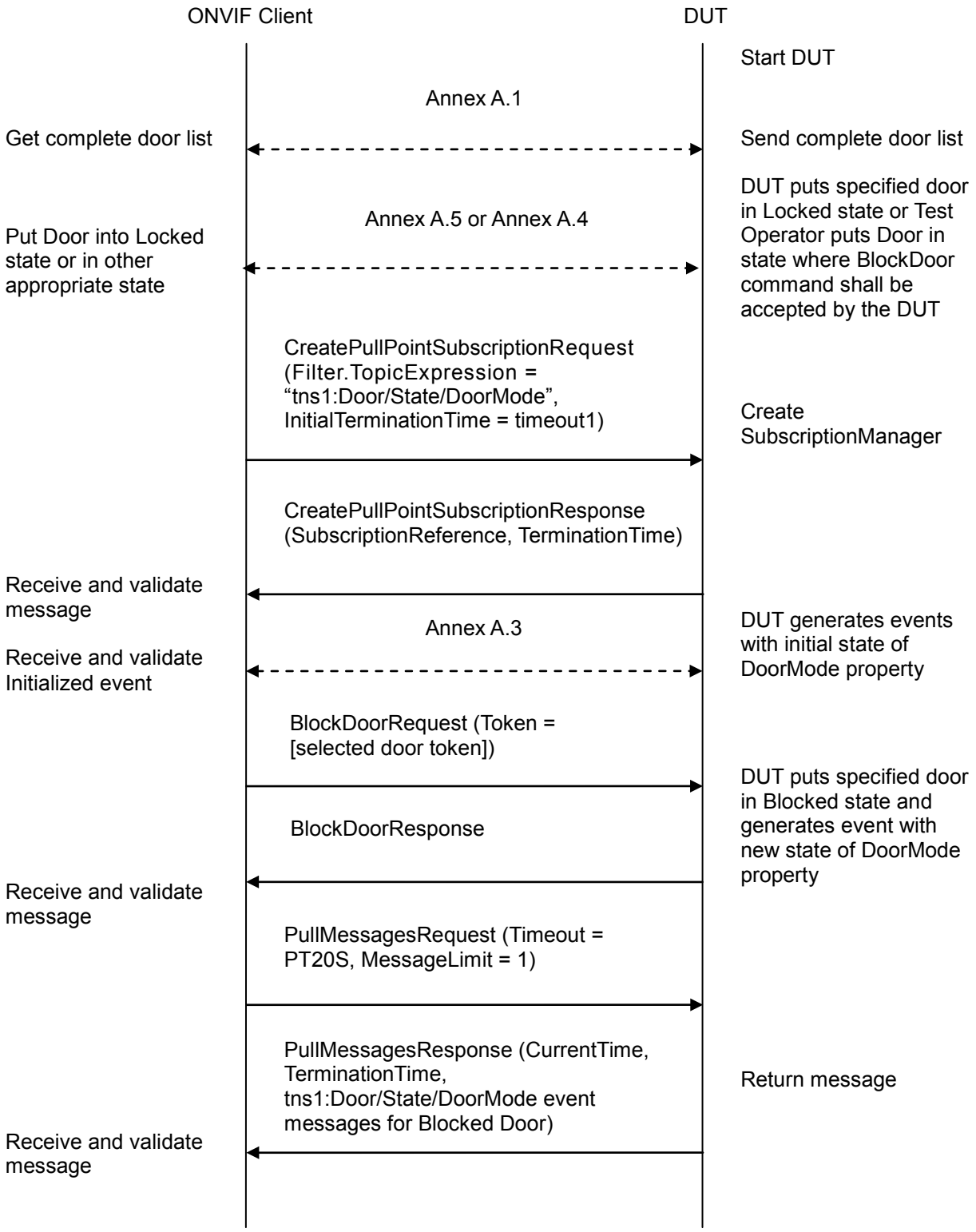
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that Block Door is accepted. To verify Door state change after Block Door command. To verify tns1:Door/State/DoorMode property events.

Pre-Requisite: Door Control Service was received from the DUT. At least one Door with Block capability is configured and added to the DUT. The doors are closed and there are no schedules.

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. ONVIF Client selects one door from the complete list of doors at step 3 with **DoorInfo.Capabilities.Block** equal to **true** and DoorInfo.Capabilities.Locked equal to **true**. If there is no Door with **DoorInfo.Capabilities.Block** equal to **true**, fail the test and skip other steps. If there is no Door with **DoorInfo.Capabilities.Block** equal to **true** and **DoorInfo.Capabilities.Lock** equal to **true,** then ONVIF Client executes Annex A.4 to start user interaction process, and then goes to step 6**.**
5. ONVIF Client will execute Annex A.5 to switch the selected door into Locked state.
6. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
8. Execute Annex A.3 to catch Initialized event for selected door.
9. ONVIF Client will invoke BlockDoorRequest message (**Token = [selected Door token]**) to change door state.
10. Verify the BlockDoorResponse message from the DUT.
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 NotificationMessages. Repeat step 11 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
13. Check that all Notification messages with PropertyOperation = "Changed" contain Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token.
14. Check that all Notification messages with PropertyOperation = "Changed" have TopicExpression equal to tns1:Door/State/DoorMode.
15. Verify that the last Notification message with PropertyOperation = "Changed" contains Data.SimpleItem item with Name = «State» and Value equal to “Blocked”. Check that other Notification messages contain Data.SimpleItem item with Name = «State» and Value with tdc:DoorMode type.
16. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid BlockDoorResponse message.

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not return at least one Door at step 3.

The DUT sent an invalid Notification messages with PropertyOperation = "Changed" (correct value for UTC time, TopicExpression and wsnt:Message, Topic = “tns1:Door/State/DoorMode”, contains Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token, contains Data.SimpleItem item with Name = «State» and Value with type equal to tdc:DoorMode) at step 12.

The DUT sent the last Notification message with PropertyOperation = "Changed" with State value other than “Blocked” at step 12.

Note: If the DUT does not return any door for step 3, skip steps 4-16, fail the test and go to the next test.

Note: If the DUT sends SOAP fault to BlockDoorRequest message then the ONVIF Client deletes Subscription Manager, starts user interaction process (see Annex A.4), and then runs the test from step 6.

Note: If in the Annex A.1A.5 the DUT sends SOAP fault to request to change door mode, then the ONVIF Client starts user interaction process (see Annex A.4), and then runs the test from step 6.

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

Note: 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.

4.3.21 DOUBLE LOCK DOOR

Test Label: Double Lock Door Verification.

Test Case ID: DOORCONTROL-3-1-30

ONVIF Core Specification Coverage: DoubleLockDoor (ONVIF Door Control Service Specification)

Command under test: DoubleLockDoor

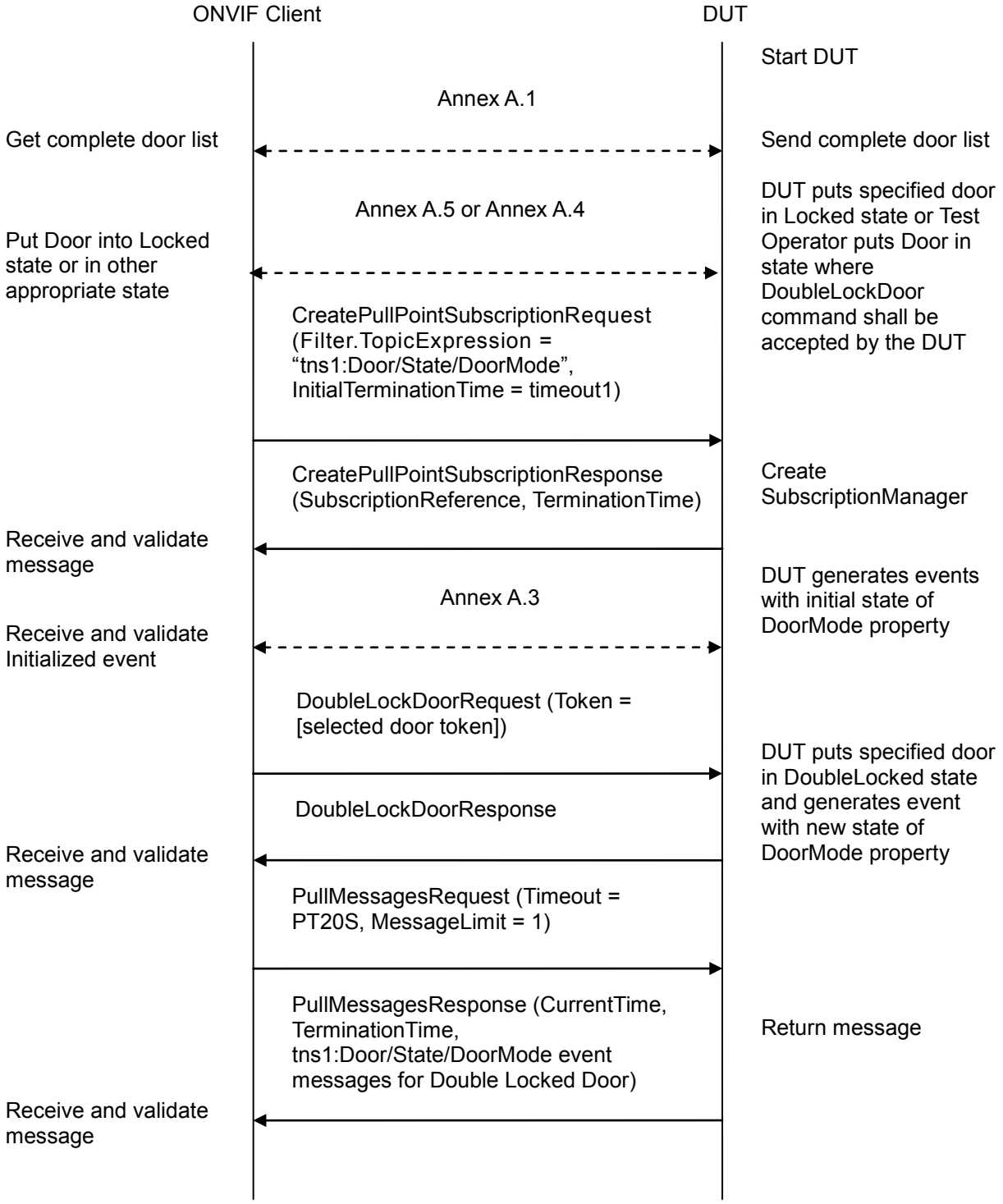
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that Double Lock Door is accepted. To verify Door state change after Double Lock Door command. To verify tns1:Door/State/DoorMode property events.

Pre-Requisite: Door Control Service was received from the DUT. At least one Door with Double Lock capability is configured and added to the DUT. The doors are closed and there are no schedules.

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

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

3. Get complete list of doors from the DUT (see Annex A.1).
4. ONVIF Client selects one door from the complete list of doors at step 3 with **DoorInfo.Capabilities.DoubleLock** equal to **true** and DoorInfo.Capabilities.Locked equal to **true**. If there is no Door with **DoorInfo.Capabilities.DoubleLock** equal to **true**, fail the test and skip other steps. If there is no Door with **DoorInfo.Capabilities.DoubleLock** equal to **true** and **DoorInfo.Capabilities.Lock** equal to **true,** then ONVIF Client executes Annex A.4 to start user interaction process and then goes to step 6.
5. ONVIF Client will execute Annex A.5 to switch the selected door into Locked state.
6. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
8. Execute Annex A.3 to catch Initialized event for selected door.
9. ONVIF Client will invoke DoubleLockDoorRequest message (**Token = [selected Door token]**) to change door state.
10. Verify the DoubleLockDoorResponse message from the DUT.
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 NotificationMessages. Repeat step 11 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
13. Check that all Notification messages with PropertyOperation = "Changed" contain Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token.
14. Check that all Notification messages with PropertyOperation = "Changed" have TopicExpression equal to tns1:Door/State/DoorMode.
15. Verify that the last Notification message with PropertyOperation = "Changed" contains Data.SimpleItem item with Name = «State» and Value equal to “DoubleLocked”. Check that other Notification messages contain Data.SimpleItem item with Name = «State» and Value with tdc:DoorMode type.
16. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid DoubleLockDoorResponse message.

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not return at least one Door at step 3.

The DUT sent an invalid Notification messages with PropertyOperation = "Changed" (correct value for UTC time, TopicExpression and wsnt:Message, Topic =

"tns1:Door/State/DoorMode", contains Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token, contains Data.SimpleItem item with Name = «State» and Value with type equal to tdc:DoorMode) at step 11.

The DUT sent the last Notification message with PropertyOperation = "Changed" with State value other than "DoubleLocked" at step 11.

Note: If the DUT does not return any door for step 3, skip steps 4-16, fail the test and go to the next test.

Note: If the DUT sends SOAP fault to DoubleLockDoorRequest message then the ONVIF Client deletes Subscription Manager, starts user interaction process (see Annex A.4,) and then runs the test from step 6.

Note: If in the Annex A.5 the DUT sends SOAP fault to request to change door mode, then the ONVIF Client starts user interaction process (see Annex A.4), and then runs the test from step 6.

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

Note: 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.

4.3.22 LOCK DOOR

Test Label: Lock Door Verification.

Test Case ID: DOORCONTROL-3-1-31

ONVIF Core Specification Coverage: LockDoor (ONVIF Door Control Service Specification)

Command under test: LockDoor

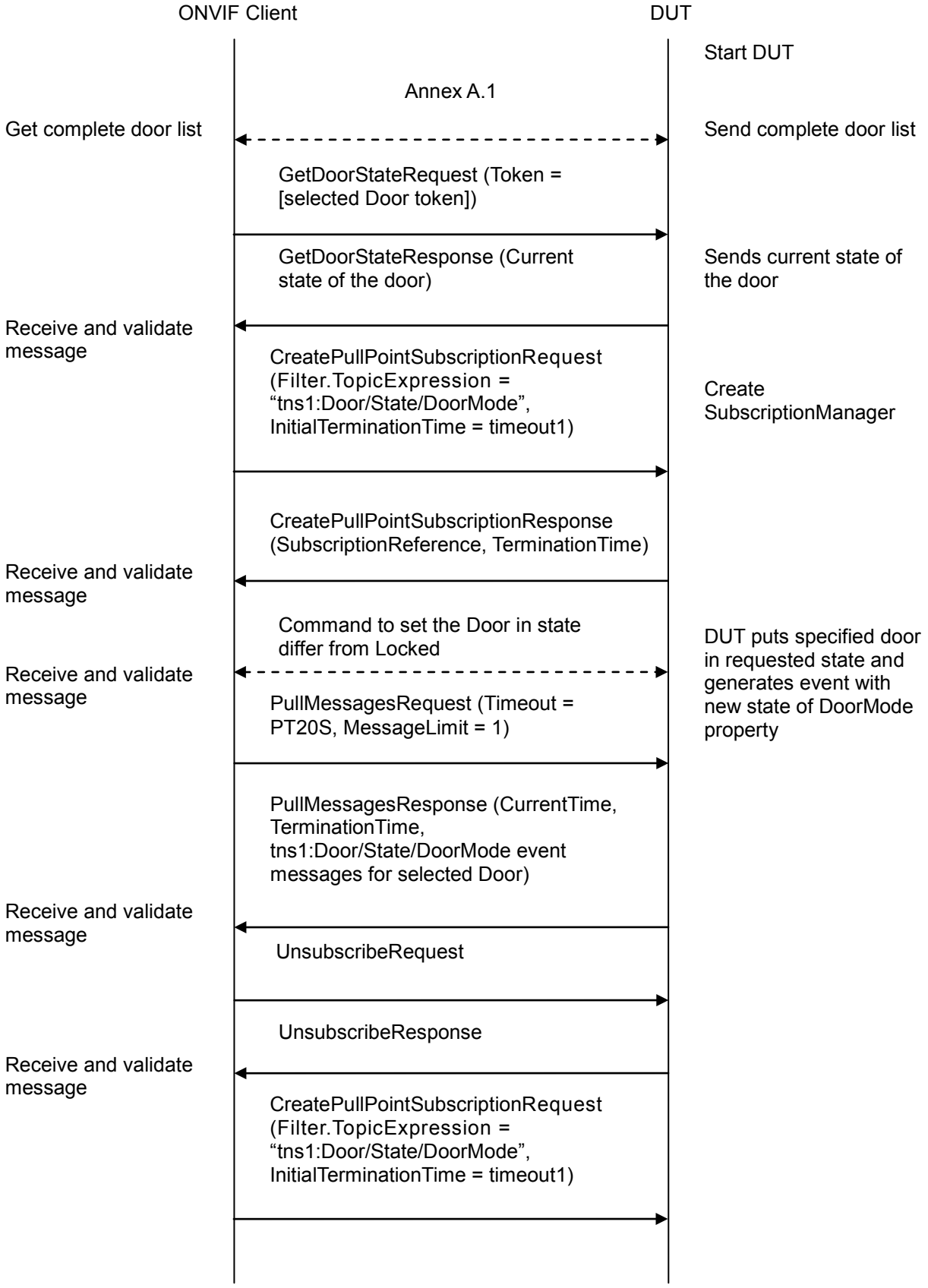
WSDL Reference: doorcontrol.wsdl

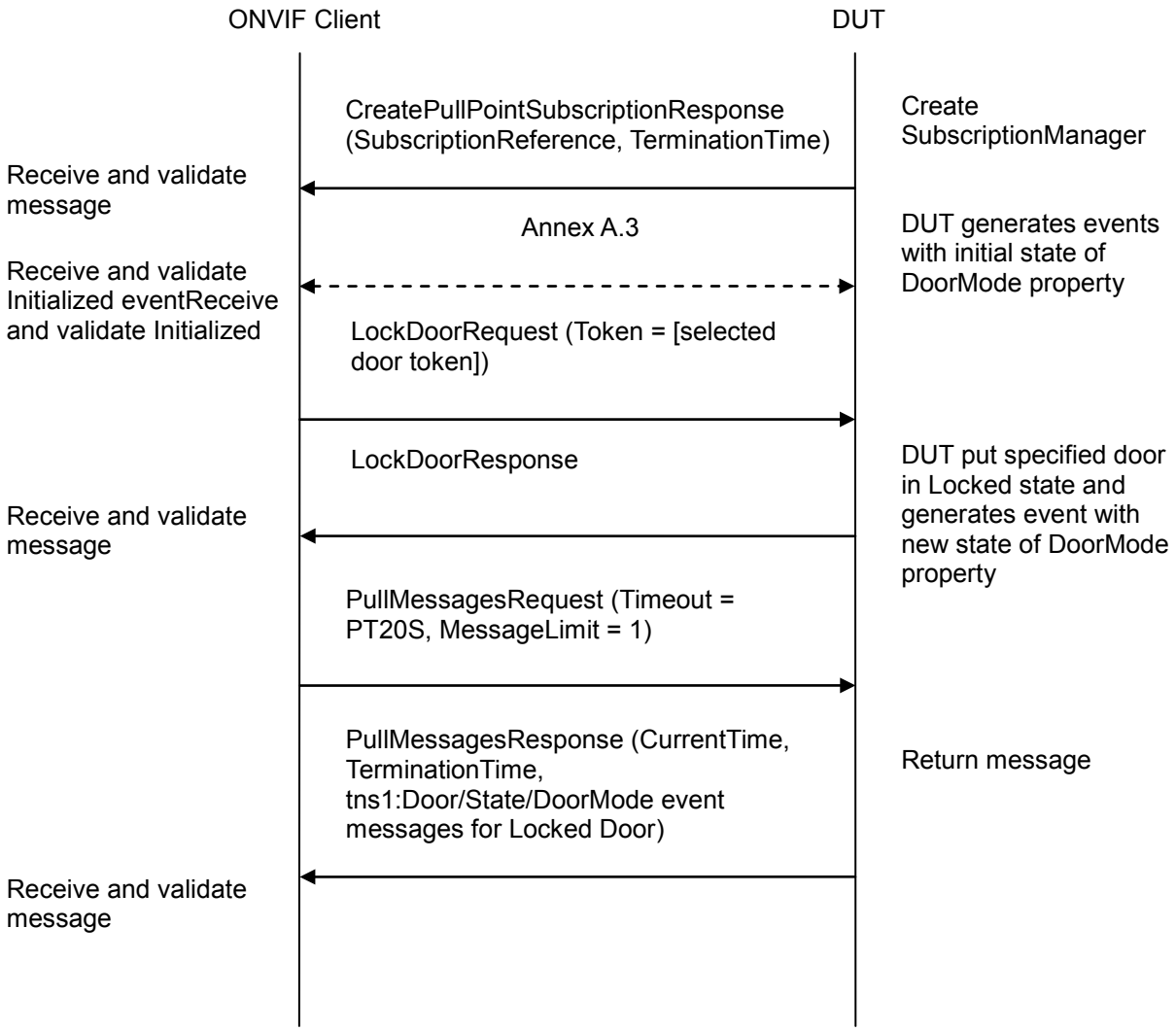
Test Purpose: To verify that Lock Door is accepted. To verify Door state change after Lock Door command. To verify tns1:Door/State/DoorMode property events.

Pre-Requisite: Door Control Service was received from the DUT. At least one Door with Lock capability is configured and added to the DUT. The doors are closed and there are no schedules.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete door list
Send complete door list
GetDoorStateRequest (Token = [selected Door token])
GetDoorStateResponse (Current state of the door)
Sends current state of the door
Receive and validate message
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/DoorMode”, InitialTerminationTime = timeout1)
Create SubscriptionManager
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Receive and validate message
Command to set the Door in state differ from Locked
DUT puts specified door in requested state and generates event with new state of DoorMode property
Receive and validate message
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorMode event messages for selected Door)
Receive and validate message
UnsubscribeRequest
UnsubscribeResponse
Receive and validate message
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/DoorMode”, InitialTerminationTime = timeout1)



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. ONVIF Client selects one door from the complete list of doors at step 3 with **DoorInfo.Capabilities.Lock** equal to true. If there is no such Door, fail the test and skip other steps.
5. ONVIF Client will invoke GetDoorStateRequest message (**Token = [selected Door token]**) to retrieve current state of the door.
6. Verify GetDoorStateResponse from the DUT.
7. If **DoorState.DoorMode** is equal to **Unlocked** or **Blocked** or **DoubleLocked or Accessed** then go to step 23.

8. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.
9. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
10. If **DoorState.DoorMode** is equal to **LockedDown** then ONVIF Client will invoke LockDownReleaseDoorRequest message (Token = “Token1”). Otherwise, go to step 15.
11. Verify LockDownReleaseDoorResponse from the DUT.
12. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
13. Verify that the DUT sends a PullMessagesResponse. Repeat step 12 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
14. Check that the last Notification message with PropertyOperation = "Changed" contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to **Token1** and contains Data.SimpleItem item with Name = «State» and Value equals to “**Locked**”.
15. If **DoorState.DoorMode** is equal to **Locked** then ONVIF Client will invoke UnlockDoorRequest message (Token = “Token1”) to move the door into state different from Locked. Otherwise, go to step 17.
16. Verify UnlockDoorResponse message from the DUT.
17. If **DoorState.DoorMode** is equal to **LockedOpen** then ONVIF Client will invoke LockOpenReleaseRequest message (Token = “Token1”) to move the door into Unlocked state.
18. Verify LockOpenReleaseResponse message from the DUT.
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. Repeat step 19 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
21. Check that the last Notification message with PropertyOperation = "Changed" contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to **Token1** and contains Data.SimpleItem item with Name = «State» and Value differ from “**Locked**”.
22. ONVIF Client deletes Subscription Manager either by calling unsubscribe or through a timeout and returns to the test.
23. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.
24. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
25. Execute Annex A.3 to catch Initialized event for selected door.
26. ONVIF Client will invoke LockDoorRequest message (**Token = [selected Door token]**) to change door state.
27. Verify the LockDoorResponse message from the DUT.
28. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

29. Verify that the DUT sends a PullMessagesResponse that contains NotificationMessages. Repeat step 28 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
30. Check that all Notification messages with PropertyOperation = "Changed" contain Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token.
31. Check that all Notification messages with PropertyOperation = "Changed" have TopicExpression equal to tns1:Door/State/DoorMode.
32. Verify that the last Notification message with PropertyOperation = "Changed" contains Data.SimpleItem item with Name = «State» and Value equal to “Locked”. Check that other Notification messages contain Data.SimpleItem item with Name = «State» and Value with tdc:DoorMode type.
33. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid GetDoorStateResponse message.

The DUT did not send valid LockDoorResponse message.

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not return at least one Door at step 3.

The DUT sent an invalid Notification messages with PropertyOperation = "Changed" (correct value for UTC time, TopicExpression and wsnt:Message, Topic = “tns1:Door/State/DoorMode”, contains Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token, contains Data.SimpleItem item with Name = «State» and Value with type equal to tdc:DoorMode) at step 29.

The DUT sent the last Notification message with PropertyOperation = "Changed" with State value other than “Locked” at step 29.

Note: If the DUT does not return any door for step 3, skip steps 4-33, fail the test and go to the next test.

Note: If the DUT sends SOAP fault to command changing the DoorState.DoorMode at step 11 or 16 or 18 or 27 then the ONVIF Client starts user interaction process (see Annex A.4), and runs the test from step 23.

Note: If the DUT sends the last notification message with door state differerent from “Locked” at step 13, then the ONVIF Client starts user interaction process (see Annex A.4), and runs the test from step 23.

Note: If the DUT sends the last notification message with door state not different from “Locked” at step 20, then the ONVIF Client starts user interaction process (see Annex A.4), and runs the test from step 23.

Note: If in the Annex A.5 the DUT sends SOAP fault to request to change door mode, then the

ONVIF Client starts user interaction process (see Annex A.4), and then runs the test from step 23.

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.

4.3.23 UNLOCK DOOR

Test Label: Unlock Door Verification.

Test Case ID: DOORCONTROL-3-1-32

ONVIF Core Specification Coverage: UnlockDoor (ONVIF Door Control Service Specification)

Command under test: UnlockDoor

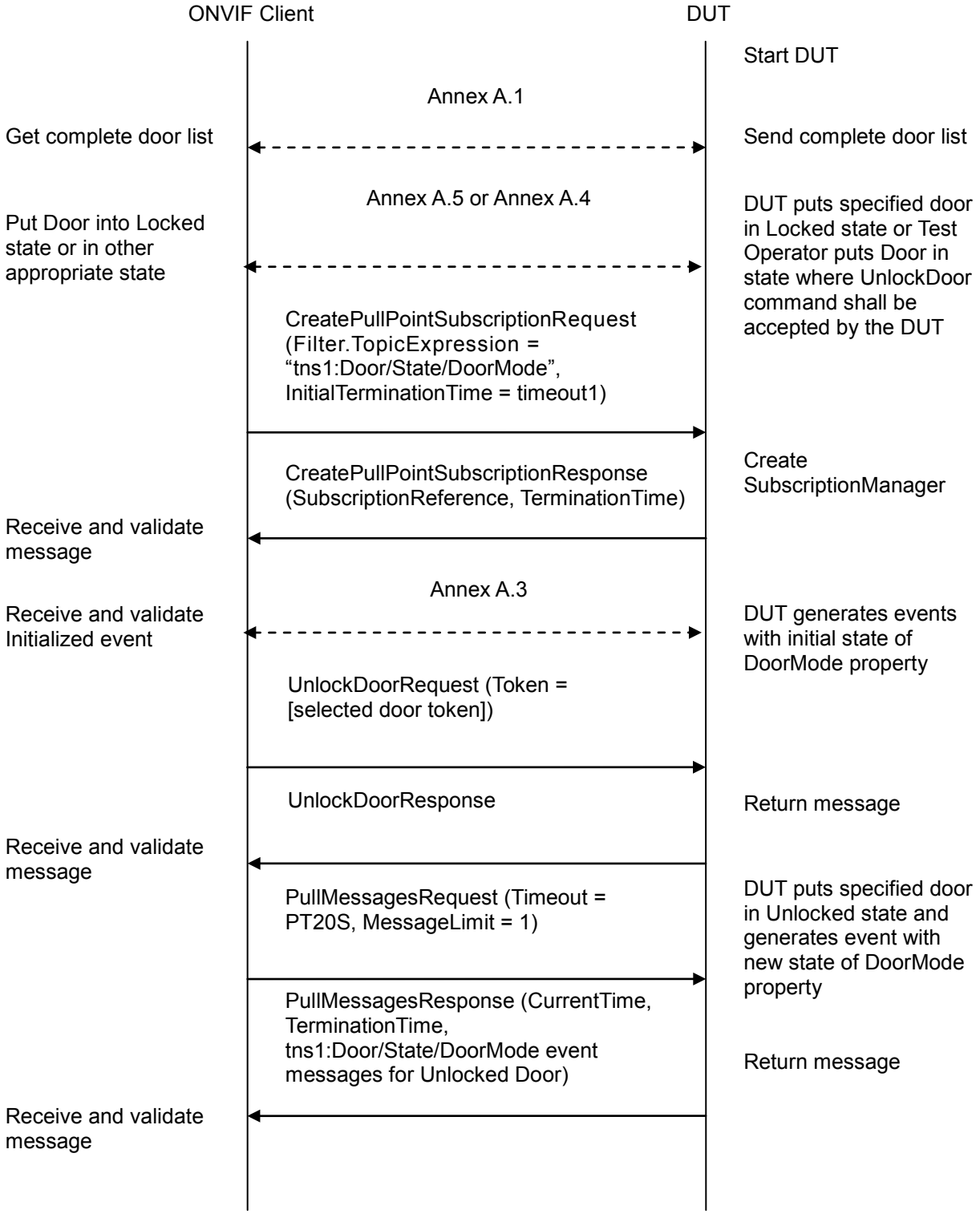
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that Unlock Door is accepted or fault is returned. To verify Door state change after Unlock Door command. To verify tns1:Door/State/DoorMode property events.

Pre-Requisite: Door Control Service was received from the DUT. At least one Door with Unlock capability is configured and added to the DUT. The doors are closed and there are no schedules.

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

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

3. Get a complete list of doors from the DUT (see Annex A.1).
4. ONVIF Client selects one door from the complete list of doors at step 3 with **DoorInfo.Capabilities.Unlock** equal to **true** and DoorInfo.Capabilities.Locked equal to **true**. If there is no Door with **DoorInfo.Capabilities.Unlock** equal to **true**, fail the test and skip other steps. If there is no Door with DoorInfo.Capabilities.Unlock equal to **true** and DoorInfo.Capabilities.Lock equal to true then ONVIF Client executes Annex A.4 to start user interaction process and then goes to step 6.
5. ONVIF Client will execute Annex A.5 to switch the selected door into Locked state.
6. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
8. Execute Annex A.3 to catch Initialized event for selected door.
9. ONVIF Client will invoke UnlockDoorRequest message (**Token = [selected Door token]**) to change door state.
10. Verify the UnlockDoorResponse message from the DUT.
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 NotificationMessages. Repeat step 11 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
13. Check that all Notification messages with PropertyOperation = "Changed" contain Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token.
14. Check that all Notification messages with PropertyOperation = "Changed" have TopicExpression equal to tns1:Door/State/DoorMode.
15. Verify that the last Notification message with PropertyOperation = "Changed" contains Data.SimpleItem item with Name = «State» and Value equal to “Unlocked”. Check that other Notification messages contain Data.SimpleItem item with Name = «State» and Value with tdc:DoorMode type.
16. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid UnlockDoorResponse message.

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not return at least one Door at step 3.

The DUT sent an invalid Notification messages with PropertyOperation = "Changed" (correct value for UTC time, TopicExpression and wsnt:Message, Topic =

"tns1:Door/State/DoorMode", contains Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token, contains Data.SimpleItem item with Name = «State» and Value with type equal to tdc:DoorMode) at step 12.

The DUT sent the last Notification message with PropertyOperation = "Changed" with State value other than "Unlocked" at step 12.

Note: If the DUT does not return any door for step 3, skip steps 4-16, fail the test and go to the next test.

Note: If the DUT sends SOAP fault to UnlockDoorRequest message then the ONVIF Client deletes Subscription Manager, starts user interaction process (see Annex A.4), and then runs the test from step 6.

Note: If in the Annex A.5 the DUT sends SOAP fault to request to change door mode, then the ONVIF Client starts user interaction process (see Annex A.4), and then runs the test from step 6.

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

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

Note: Test will be failed, if no required Notification messages are received for step 9 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.

4.3.24 LOCK OPEN DOOR

Test Label: Lock Open Door Verification.

Test Case ID: DOORCONTROL-3-1-33

ONVIF Core Specification: LockOpenDoor (ONVIF Door Control Service Specification)

Command under test: LockOpenDoor

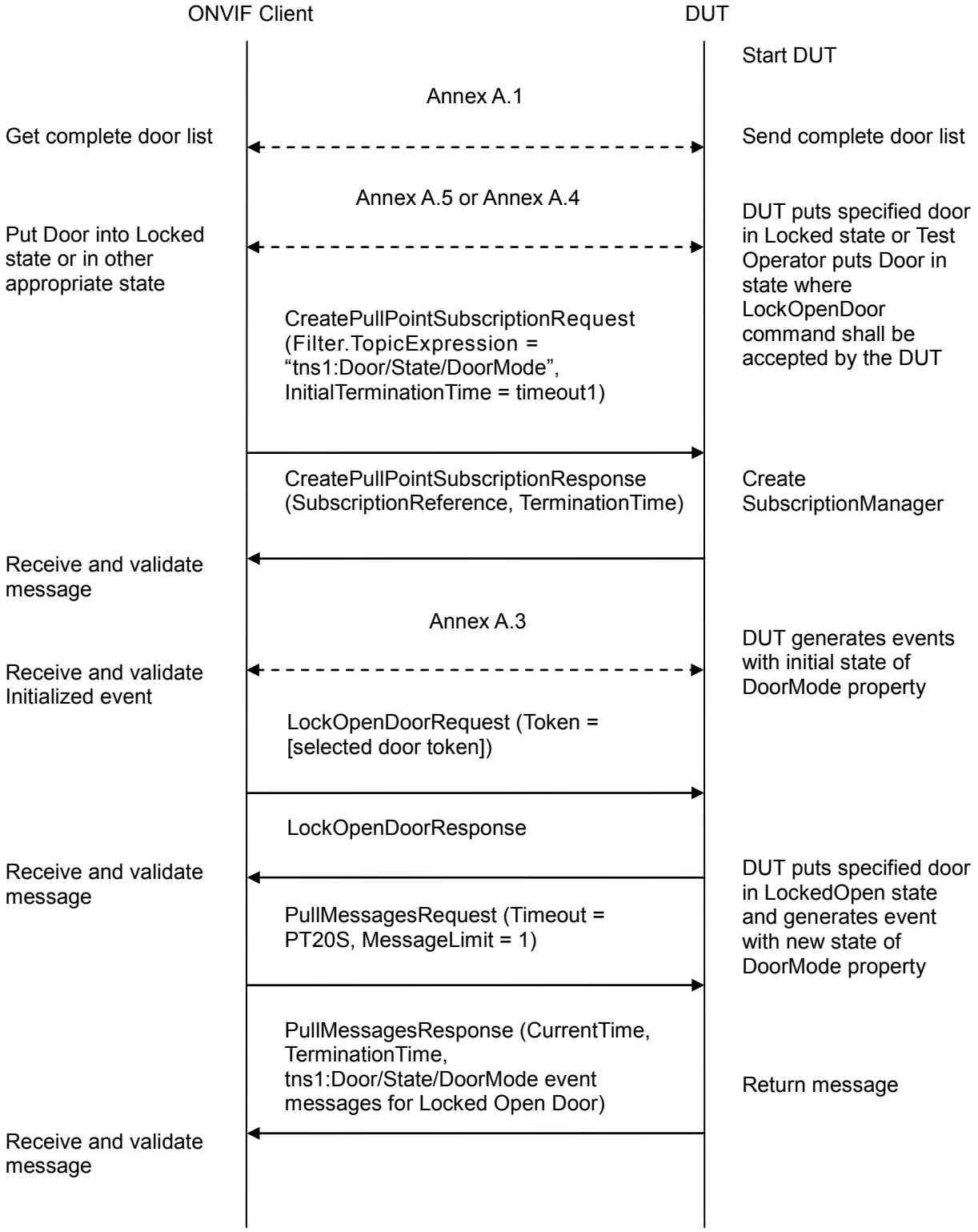
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that Lock Open Door is accepted. To verify Door state change after Lock Open Door command. To verify tns1:Door/State/DoorMode property events.

Pre-Requisite: Door Control Service was received from the DUT. At least one Door with Lock Open capability is configured and added to the DUT. The doors are closed and there are no schedules.

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. ONVIF Client selects one door from the complete list of doors at step 3 with **DoorInfo.Capabilities.LockOpen** equal to **true** and DoorInfo.Capabilities.Locked equal to **true**. If there is no Door with **DoorInfo.Capabilities.LockOpen** equal to **true**, fail the test and skip other steps. If there is no Door with DoorInfo.Capabilities.LockOpen equal to true and DoorInfo.Capabilities.Lock equal to true then ONVIF Client executes Annex A.4 to start user interaction process and then goes to step 6.
5. ONVIF Client will execute Annex A.5 to switch the selected door into Locked state.
6. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
8. Execute Annex A.3 to catch Initialized event for selected door.
9. ONVIF Client will invoke LockOpenDoorRequest message (**Token = [selected Door token]**) to change door state.
10. Verify the LockOpenDoorResponse message from the DUT.
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 NotificationMessages. Repeat step 11 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
13. Check that all Notification messages with PropertyOperation = "Changed" contain Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token.
14. Check that all Notification messages with PropertyOperation = "Changed" have TopicExpression is equal to tns1:Door/State/DoorMode.
15. Verify that the last Notification message with PropertyOperation = "Changed" contains Data.SimpleItem item with Name = «State» and Value equal to “LockedOpen”. Check that other Notification messages contain Data.SimpleItem item with Name = «State» and Value with tdc:DoorMode type.
16. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid LockOpenDoorResponse message.

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not return at least one Door at step 3.

The DUT sent an invalid Notification messages with PropertyOperation = "Changed" (correct value for UTC time, TopicExpression and wsnt:Message, Topic = “tns1:Door/State/DoorMode”, contains Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token, contains Data.SimpleItem item with Name = «State» and Value with type equal to tdc:DoorMode) at step 12.

The DUT sent the last Notification message with PropertyOperation = "Changed" with State value other than “LockedOpen” at step 12.

Note: If the DUT does not return any door for step 3, skip steps 4-16, fail the test and go to the next test.

Note: If the DUT sends SOAP fault to LockOpenDoorRequest message then the ONVIF Client deletes Subscription Manager, starts user interaction process (see Annex A.4), and then runs the test from step 6.

Note: If in the Annex A.5 the DUT sends SOAP fault to request to change door mode, then the ONVIF Client starts user interaction process (see Annex A.4), and then runs the test from step 6.

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

Note: 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.

4.3.25 LOCK OPEN RELEASE DOOR

Test Label: Lock Open Release Door Verification.

Test Case ID: DOORCONTROL-3-1-34

ONVIF Core Specification: LockOpenReleaseDoor (ONVIF Door Control Service Specification)

Command under test: LockOpenReleaseDoor

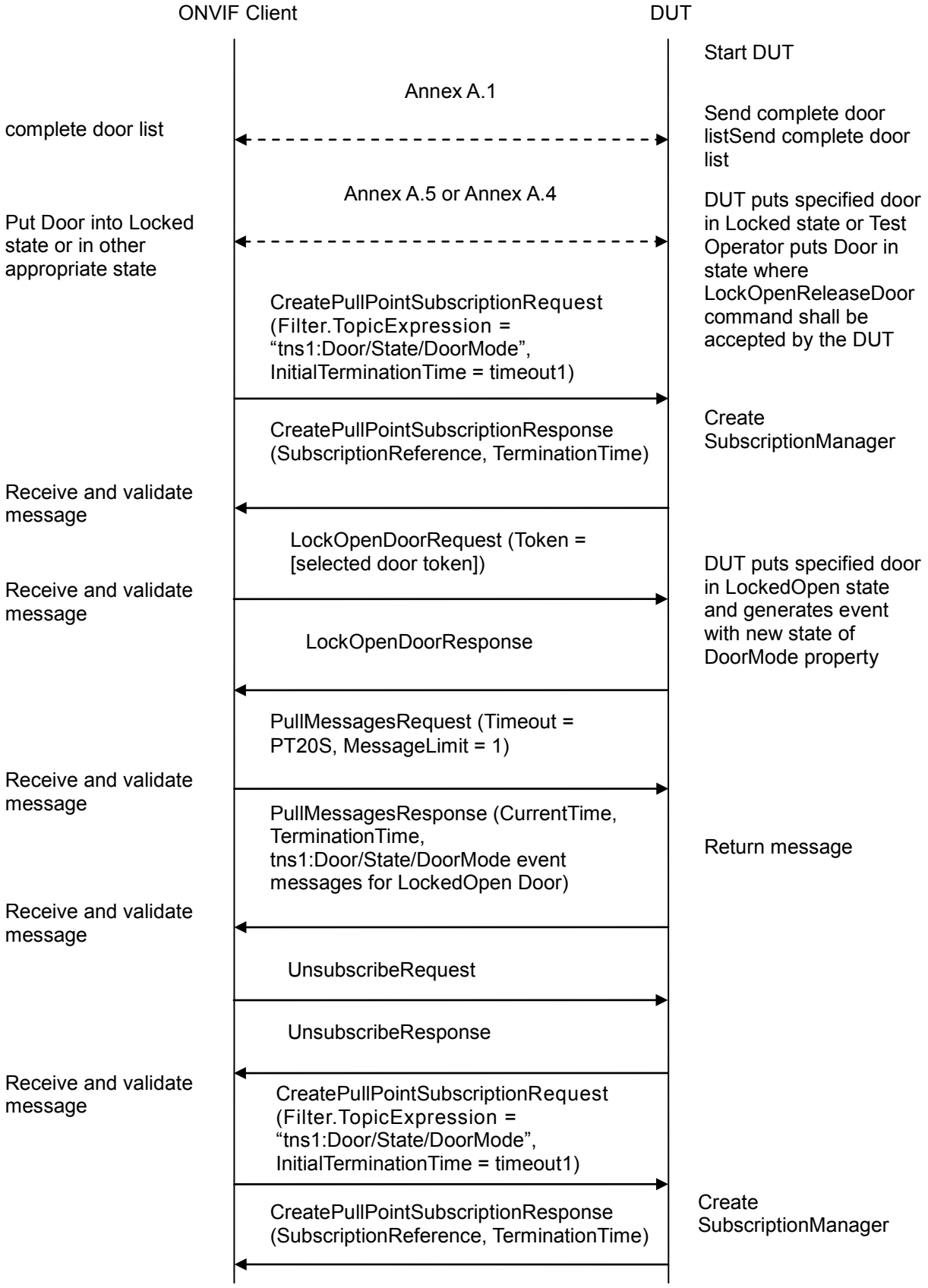
WSDL Reference: doorcontrol.wsdl

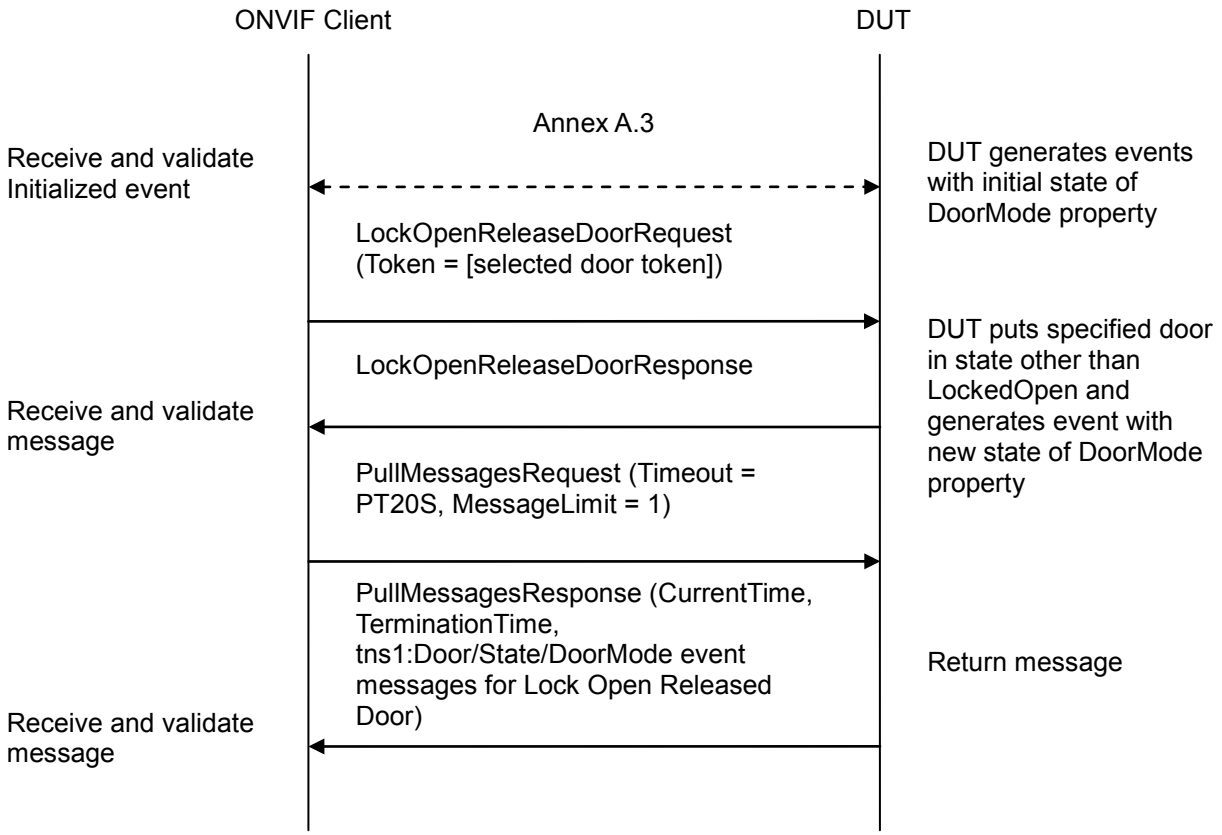
Test Purpose: To verify that Lock Open Release Door is accepted. To verify Door state change after Lock Open Release Door command. To verify tns1:Door/State/DoorMode property events.

Pre-Requisite: Door Control Service was received from the DUT. At least one Door with Lock Open capability is configured and added to the DUT. The doors are closed and there are no schedules.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
complete door list
Send complete door listSend complete door list
Annex A.5 or Annex A.4
Put Door into Locked state or in other appropriate state
DUT puts specified door in Locked state or Test Operator puts Door in state where LockOpenReleaseDoor command shall be accepted by the DUT
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/DoorMode”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
LockOpenDoorRequest (Token = [selected door token])
Receive and validate message
DUT puts specified door in LockedOpen state and generates event with new state of DoorMode property
LockOpenDoorResponse
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
Receive and validate message
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorMode event messages for LockedOpen Door)
Return message
Receive and validate message
UnsubscribeRequest
UnsubscribeResponse
Receive and validate message
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/DoorMode”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. ONVIF Client selects one door from the complete list of doors at step 3 with **DoorInfo.Capabilities.LockOpen** equal to **true** and DoorInfo.Capabilities.Locked equal to **true**. If there is no Door with **DoorInfo.Capabilities.LockOpen** equal to **true**, fail the test and skip other steps. If there is no Door with DoorInfo.Capabilities.LockOpen equal to true and DoorInfo.Capabilities.Lock equal to true then ONVIF Client executes Annex A.4 to start user interaction process and then goes to step 14.
5. ONVIF Client will invoke GetDoorStateRequest message (**Token = "Token1"**, where Token1 is selected door) to retrieve current state of the door.
6. Verify GetDoorStateResponse from the DUT. If **DoorState.DoorMode** is equal to **LockedOpen** then skip steps 7-15 and go to step 16.
7. ONVIF Client will execute Annex A.5 to switch the selected door into Locked state.
8. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.
9. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

10. ONVIF Client will invoke LockOpenDoorRequest message (**Token = [selected Door token]**) to move the door in LockedOpen state.
11. Verify LockOpenDoorResponse message from the DUT.
12. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
13. Verify that the DUT sends a PullMessagesResponse. Repeat step 12 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
14. Check that the last Notification message with PropertyOperation = "Changed" contains Source.SimpleItem item with Name = "DoorToken" and Value equal to selected door tocken and contains Data.SimpleItem item with Name = «State» and Value equal to "**LockedOpen**".
15. ONVIF Client deletes Subscription Manager either by calling unsubscribe or through a timeout.
16. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.
17. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
18. Execute Annex A.3 to catch Initialized event for selected door. State of the Door received in Annex A.3 will be assumed as initial State of this Door.
19. Check that initial state of the Door equals to "**LockedOpen**".
20. ONVIF Client will invoke LockOpenReleaseDoorRequest message (**Token = [selected Door token]**) to change door state.
21. Verify the LockOpenDoorReleaseResponse message from the DUT.
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 NotificationMessages. Repeat step 22 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
24. Check that all Notification messages with PropertyOperation = "Changed" contain Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token.
25. Check that all Notification messages with PropertyOperation = "Changed" have TopicExpression is equal to tns1:Door/State/DoorMode.
26. Virify that last Notification message with PropertyOperation = "Changed" contains Data.SimpleItem item with Name = «State» and Value other than "LockedOpen". Check that other Notification messages contain Data.SimpleItem item with Name = «State» and Value with tdc:DoorMode type.
27. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid LockOpenReleaseDoorResponse message.

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not return at least one Door at step 3.

The DUT sent the last Notification message with PropertyOperation = "Changed" with State value not equal to “LockedOpen” at step 13.

The DUT sent Notification message for selected door token with PropertyOperation = "Initialized" with State value not equal to “LockedOpen” at step 18.

The DUT sent an invalid Notification messages with PropertyOperation = "Changed" (correct value for UTC time, TopicExpression and wsnt:Message, Topic = “tns1:Door/State/DoorMode”, contains Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token, contains Data.SimpleItem item with Name = «State» and Value with type equal to tdc:DoorMode) at step 23.

The DUT sent the last Notification message with PropertyOperation = "Changed" with State value equal to “LockedOpen” at step 23.

Note: If the DUT does not return any door for step 3, skip steps 4-27, fail the test and go to the next test.

Note: If the DUT sends SOAP fault to LockOpenDoorRequest message or to LockOpenRealiseDoorRequest then the ONVIF Client deletes Subscription Manager, starts user interaction process (see Annex A.4), and then runs the test from step 16.

Note: If in the Annex A.5 the DUT sends SOAP fault to request to change door mode, then the ONVIF Client starts user interaction process (see Annex A.4), and then runs the test from step 16.

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.

4.3.26 LOCK DOWN DOOR

Test Label: Lock Down Door Verification.

Test Case ID: DOORCONTROL-3-1-35

ONVIF Core Specification: LockDownDoor (ONVIF Door Control Service Specification)

Command under test: LockDownDoor

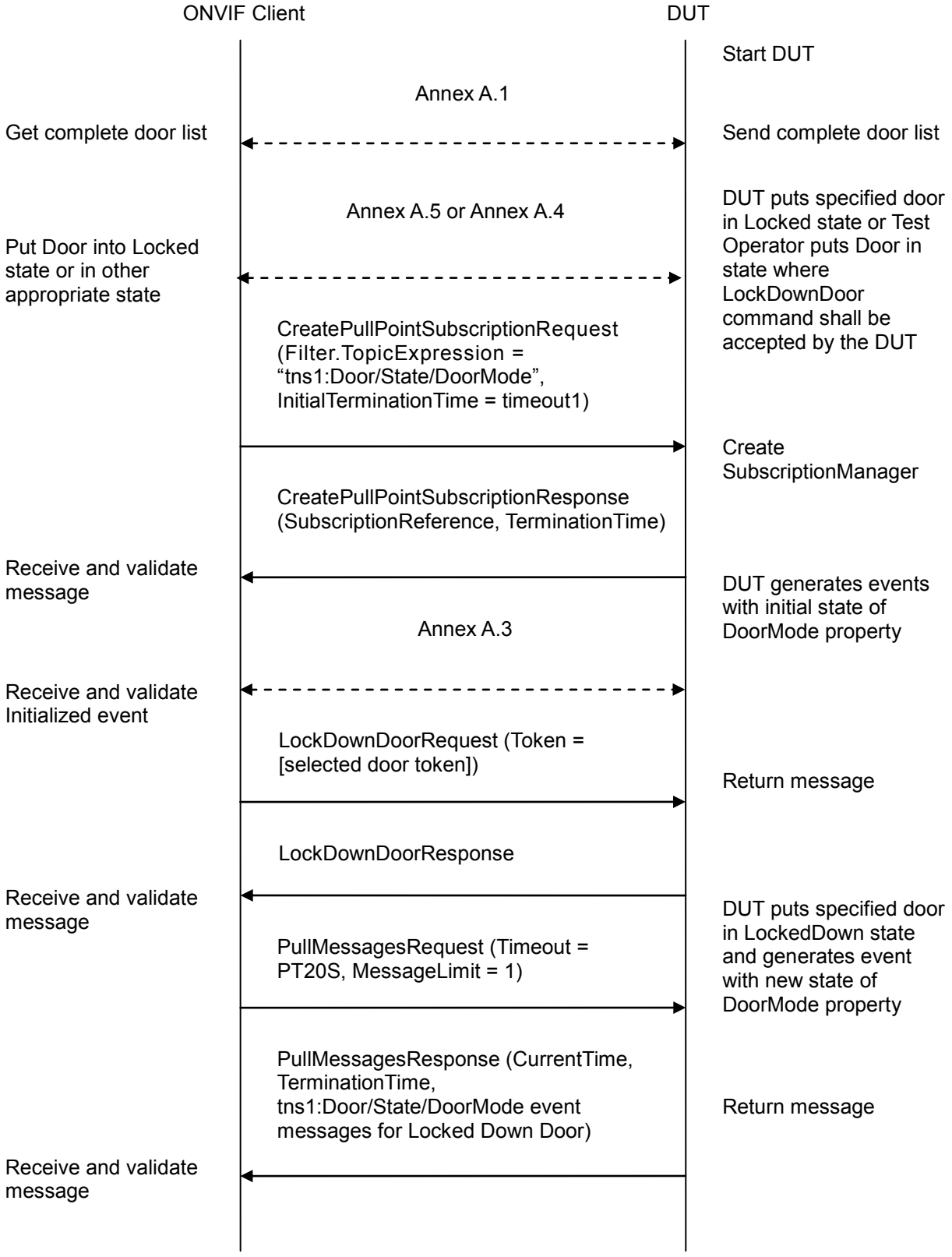
WSDL Reference: doorcontrol.wsdl

Test Purpose: To verify that Lock Down Door is accepted. To verify Door state change after Lock Down Door command. To verify tns1:Door/State/DoorMode property events.

Pre-Requisite: Door Control Service was received from the DUT. At least one Door with Lock Down capability is configured and added to the DUT. The doors are closed and there are no schedules.

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. ONVIF Client selects one door from the complete list of doors at step 3 with **DoorInfo.Capabilities.LockDown** equal to **true** and DoorInfo.Capabilities.Locked equal to **true**. If there is no Door with **DoorInfo.Capabilities.LockDown** equal to **true**, fail the test and skip other steps. If there is no Door with DoorInfo.Capabilities.LockDown equal to true and DoorInfo.Capabilities.Lock equal to true then ONVIF Client executes Annex A.4 to start user interaction process and then goes to step 6.
5. ONVIF Client will execute Annex A.5 to switch the selected door into Locked state.
6. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
8. Execute Annex A.3 to catch Initialized event for selected door.
9. ONVIF Client will invoke LockDownDoorRequest message (**Token = [selected Door token]**) to change door state.
10. Verify the LockDownDoorResponse message from the DUT.
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 NotificationMessages. Repeat step 11 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
13. Check that all Notification messages with PropertyOperation = "Changed" contain Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token.
14. Check that all Notification messages with PropertyOperation = "Changed" have TopicExpression is equal to tns1:Door/State/DoorMode.
15. Verify that the last Notification message with PropertyOperation = "Changed" contains Data.SimpleItem item with Name = «State» and Value equal to “LockedDown”. Check that other Notification messages contain Data.SimpleItem item with Name = «State» and Value with tdc:DoorMode type.
16. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid LockDownDoorResponse message.

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not return at least one Door at step 3.

The DUT sent an invalid Notification messages with PropertyOperation = "Changed" (correct value for UTC time, TopicExpression and wsnt:Message, Topic = “tns1:Door/State/DoorMode”, contains Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token, contains Data.SimpleItem item with Name = «State» and Value with type equal to tdc:DoorMode) at step 12.

The DUT sent the last Notification message with PropertyOperation = "Changed" with State value other than “LockedDown” at step 12.

Note: If the DUT does not return any door for step 3, skip steps 4-16, fail the test and go to the next test.

Note: If the DUT sends SOAP fault to LockDownDoorRequest message, then the ONVIF Client deletes Subscription Manager, starts user interaction process (see Annex A.4), and then runs the test from step 6.

Note: If in the Annex A.5 the DUT sends SOAP fault to request to change door mode, then the ONVIF Client starts user interaction process (see Annex A.4), and then runs the test from step 6.

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

Note: 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.

4.3.27 LOCK DOWN RELEASE DOOR

Test Label: Lock Down Release Door Verification.

Test Case ID: DOORCONTROL-3-1-36

ONVIF Core Specification: LockDownReleaseDoor (ONVIF Door Control Service Specification)

Command under test: LockDownReleaseDoor

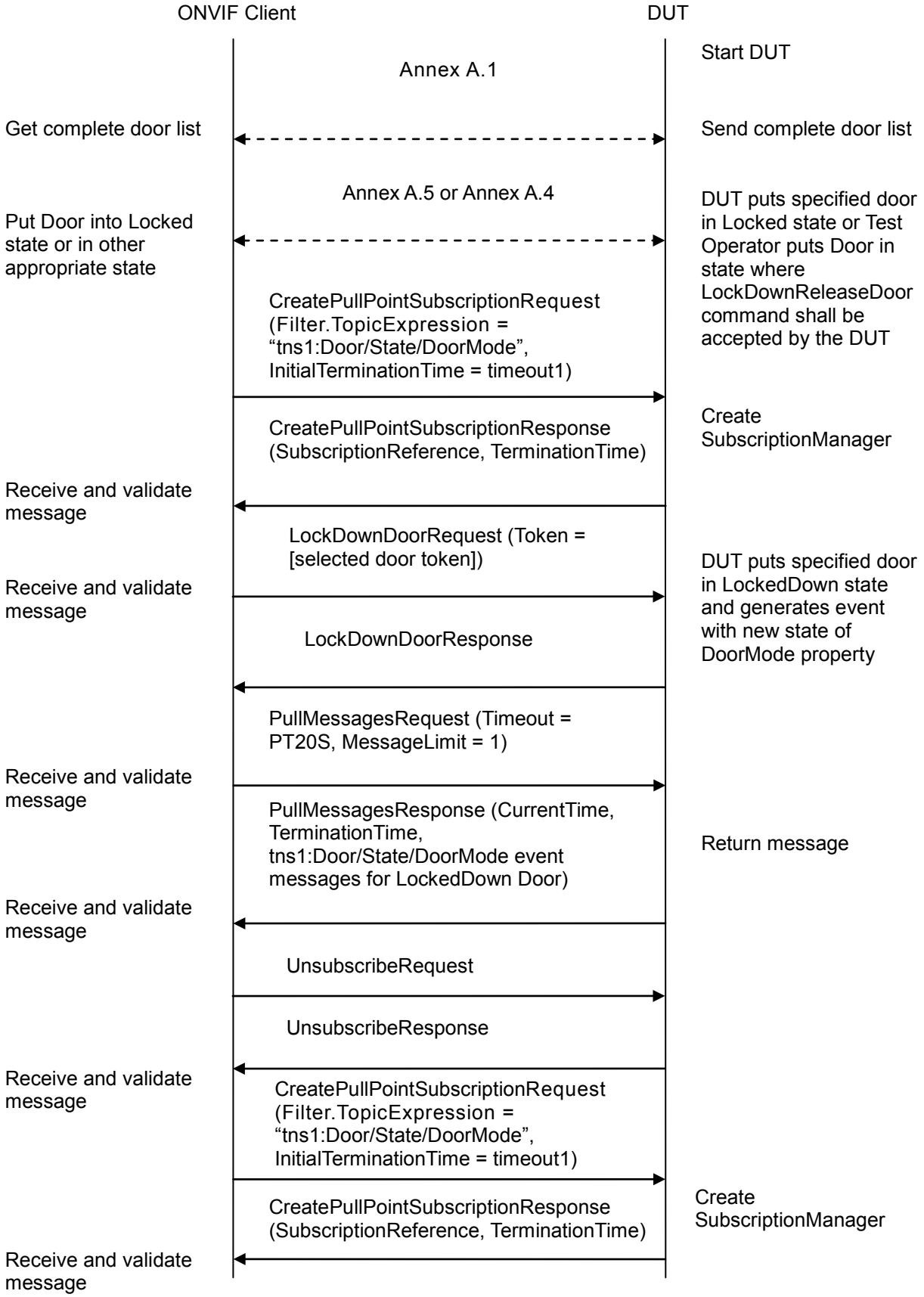
WSDL Reference: doorcontrol.wsdl

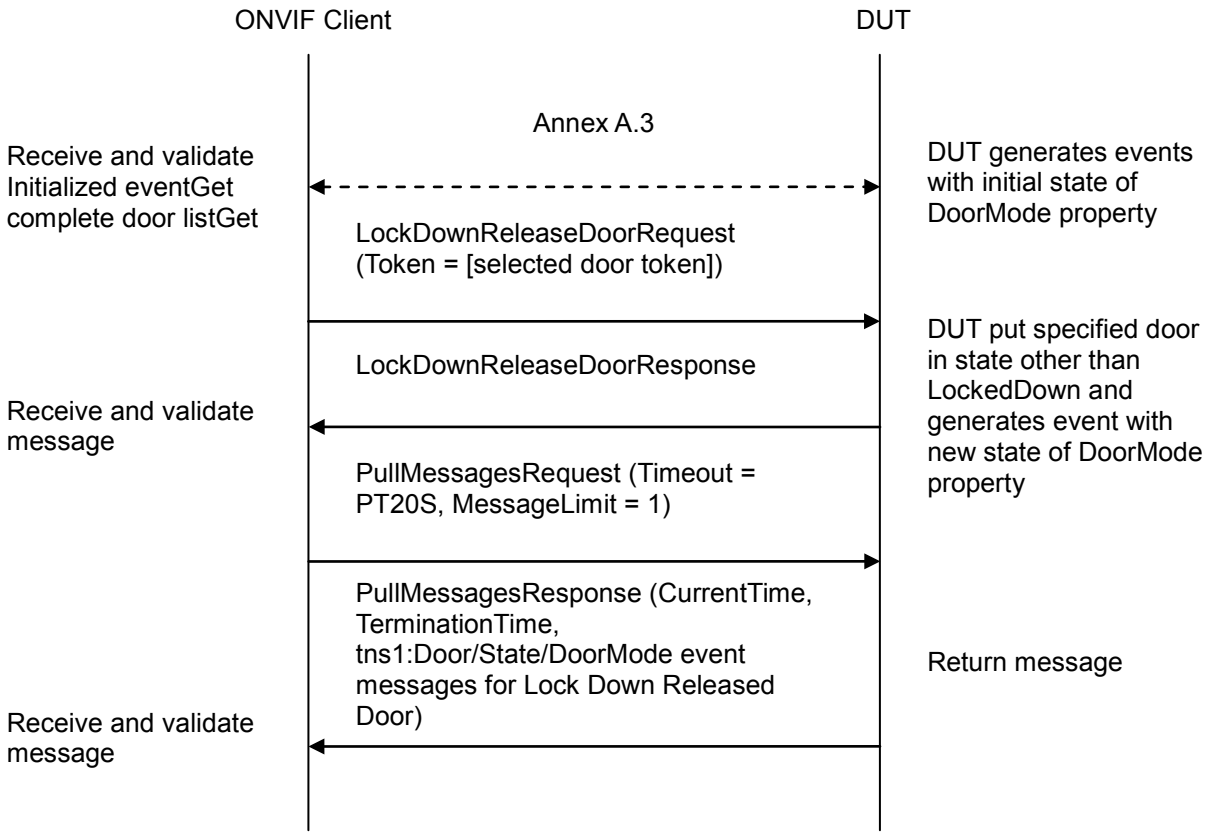
Test Purpose: To verify that Lock Down Release Door is accepted. To verify Door state change after Lock Down Release Door command. To verify tns1:Door/State/DoorMode property events.

Pre-Requisite: Door Control Service was received from the DUT. At least one Door with Lock Down capability is configured and added to the DUT. The doors are closed and there are no schedules.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete door list
Send complete door list
Annex A.5 or Annex A.4
Put Door into Locked state or in other appropriate state
DUT puts specified door in Locked state or Test Operator puts Door in state where LockDownReleaseDoor command shall be accepted by the DUT
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/DoorMode”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
LockDownDoorRequest (Token = [selected door token])
Receive and validate message
DUT puts specified door in LockedDown state and generates event with new state of DoorMode property
LockDownDoorResponse
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
Receive and validate message
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorMode event messages for LockedDown Door)
Return message
Receive and validate message
UnsubscribeRequest
UnsubscribeResponse
Receive and validate message
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/DoorMode”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. ONVIF Client selects one door from the complete list of doors at step 3 with **DoorInfo.Capabilities.LockDown** equal to **true** and DoorInfo.Capabilities.Locked equal to **true**. If there is no Door with **DoorInfo.Capabilities.LockDown** equal to **true**, fail the test and skip other steps. If there is no Door with DoorInfo.Capabilities.LockDown equal to true and DoorInfo.Capabilities.Lock equal to true then ONVIF Client executes Annex A.4 to start user interaction process and then goes to step 14.
5. ONVIF Client will invoke GetDoorStateRequest message (**Token = "Token1"**, where Token1 is selected door) to retrieve current state of the door.
6. Verify GetDoorStateResponse from the DUT. If **DoorState.DoorMode** is equal to **LockedDown** then skip steps 7-15 and go to step 16.
7. ONVIF Client will execute Annex A.5 to switch the selected door into Locked state.
8. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.
9. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

10. ONVIF Client will invoke LockDownDoorRequest message (**Token = [selected Door token]**) to move the door in LockedDown state.
11. Verify LockDownDoorResponse message from the DUT.
12. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
13. Verify that the DUT sends a PullMessagesResponse. Repeat step 12 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
14. Verify that the last Notification message with PropertyOperation = "Changed" contains Source.SimpleItem item with Name = "DoorToken" and Value equal to selected door tocken and contains Data.SimpleItem item with Name = «State» and Value equal to "**LockedDown**".
15. ONVIF Client deletes Subscription Manager either by calling unsubscribe or through a timeout.
16. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.
17. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
18. Execute Annex A.3 to catch Initialized event for selected door. State of the Door received in Annex A.3 will be assumed as initial State of this Door.
19. Check that initial state of the Door equals to "**LockedDown**".
20. ONVIF Client will invoke LockDownReleaseDoorRequest message (**Token = [selected Door token]**) to change door state.
21. Verify the LockDownReleaseDoorResponse message from the DUT.
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 NotificationMessages. Repeat step 22 until timeout has expired (Operation Delay option from ONVIF Device Test Tool will be used).
24. Check that all Notification messages with PropertyOperation = "Changed" contain Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token.
25. Check that all Notification messages with PropertyOperation = "Changed" have TopicExpression is equal to tns1:Door/State/DoorMode.
26. Verify that the last Notification message with PropertyOperation = "Changed" contains Data.SimpleItem item with Name = «State» and Value other than "LockedDown". Check that other Notification messages contain Data.SimpleItem item with Name = «State» and Value with tdc:DoorMode type.
27. Verify received Notification messages (correct value for UTC time, TopicExpression and wsnt:Message).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send valid LockDownReleaseDoorResponse message.

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not return at least one Door at step 3.

The DUT sent the last Notification message with PropertyOperation = "Changed" with State value not equal to “LockedDown” at step 13.

The DUT sent Notification message for selected door token with PropertyOperation = "Initialized" with State value not equal to “LockedDown” at step 18.

The DUT sent an invalid Notification messages with PropertyOperation = "Changed" (correct value for UTC time, TopicExpression and wsnt:Message, Topic = “tns1:Door/State/DoorMode”, contains Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token, contains Data.SimpleItem item with Name = «State» and Value with type equal to tdc:DoorMode) at step 23.

The DUT sent the last Notification message with PropertyOperation = "Changed" with State value equal to “LockedDown” at step 23.

Note: If the DUT does not return any door for step 3, skip steps 4-27, fail the test and go to the next test.

Note: If the DUT sends SOAP fault to LockDownDoorRequest message or to LockDownReleaseDoorRequest then the ONVIF Client deletes Subscription Manager, starts user interaction process (see Annex A.4), and then runs the test from step 16.

Note: If in the Annex A.5 the DUT sends SOAP fault to request to change door mode, then the ONVIF Client starts user interaction process (see Annex A.4), and then runs the test from step 16.

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.

4.4 Consistency

4.4.1 GET ACCESS POINT INFO LIST AND GET DOOR INFO LIST CONSISTENCY

Test Label: Get Door Info List and Get Access Point Info List Consistency Verification.

Test Case ID: DOORCONTROL-5-1-1

ONVIF Core Specification Coverage: GetDoorInfoList (ONVIF Door Control Service Specification), GetAccessPointInfoList (ONVIF Access Control Service Specification)

Command under test: GetDoorInfoList, GetAccessPointInfoList

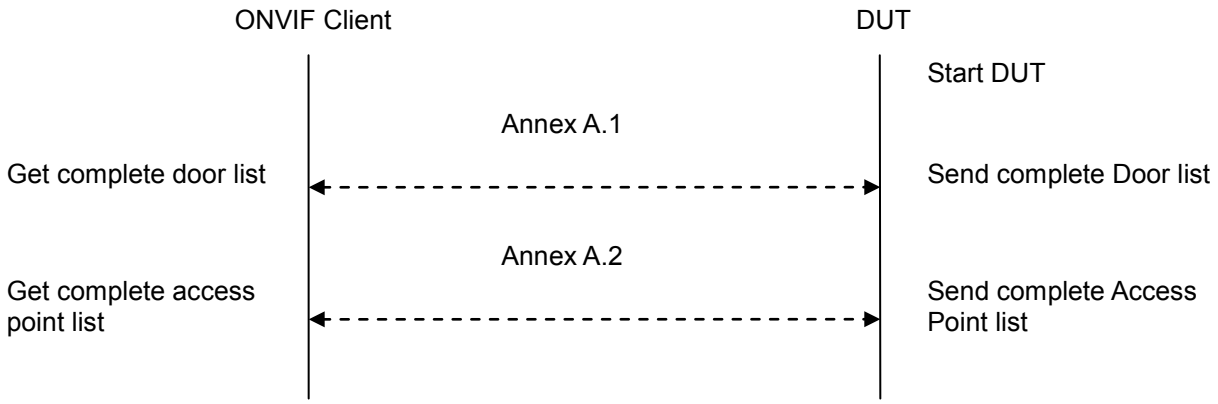
WSDL Reference: accesscontrol.wsdl, doorcontrol.wsdl

Test Purpose: To verify that Door Info List contains all Doors from Access Point Info List.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1)
4. Get a complete list of access points from the DUT (see Annex A.2)
5. Verify that the complete list of doors contains all Doors included in the complete list of access points in Entity element in case Type is skipped or equal to tdc:Door.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT returned at least one Door listed in Entity element of AccessPointInfo listed in the complete list of access points in case Type is skipped or equal to tdc:Door that was not included in the complete list of doors.

The DUT did not return at least one Door at step 3.

The DUT did not return at least one Access Point at step 4.

4.5 Property Events

4.5.1 DOOR CONTROL – DOOR MODE EVENT

Test Label: Door Control Service Property Events Check (Door Mode).

Test Case ID: DOORCONTROL-6-1-1

ONVIF Core Specification Coverage: tns1:Door/State/DoorMode (ONVIF Door Control Service Specification), GetDoorState (ONVIF Door Control Service Specification), Properties (ONVIF Core Specification)

Command under test: GetDoorState

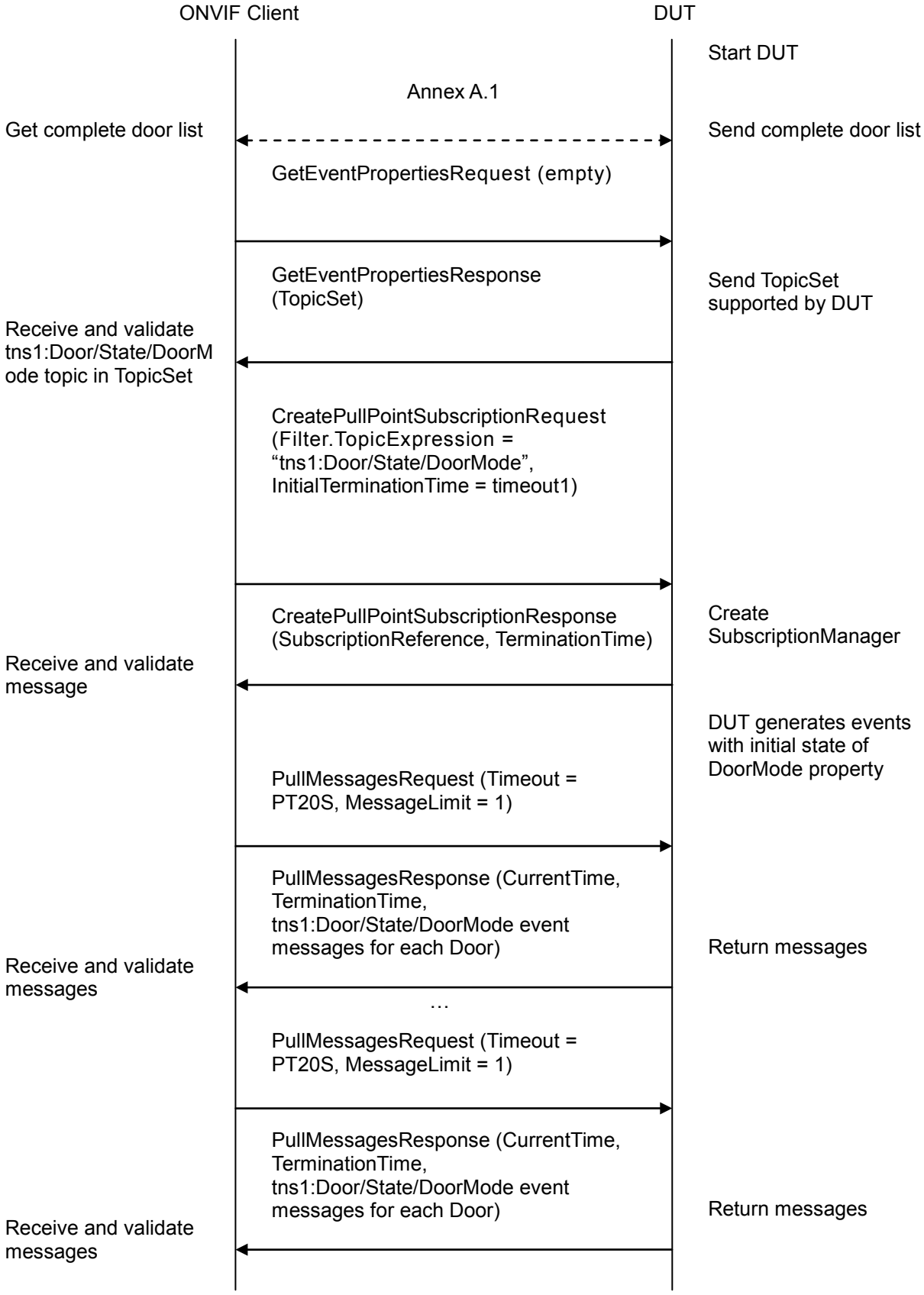
WSDL Reference: event.wsdl, doorcontrol.wsdl

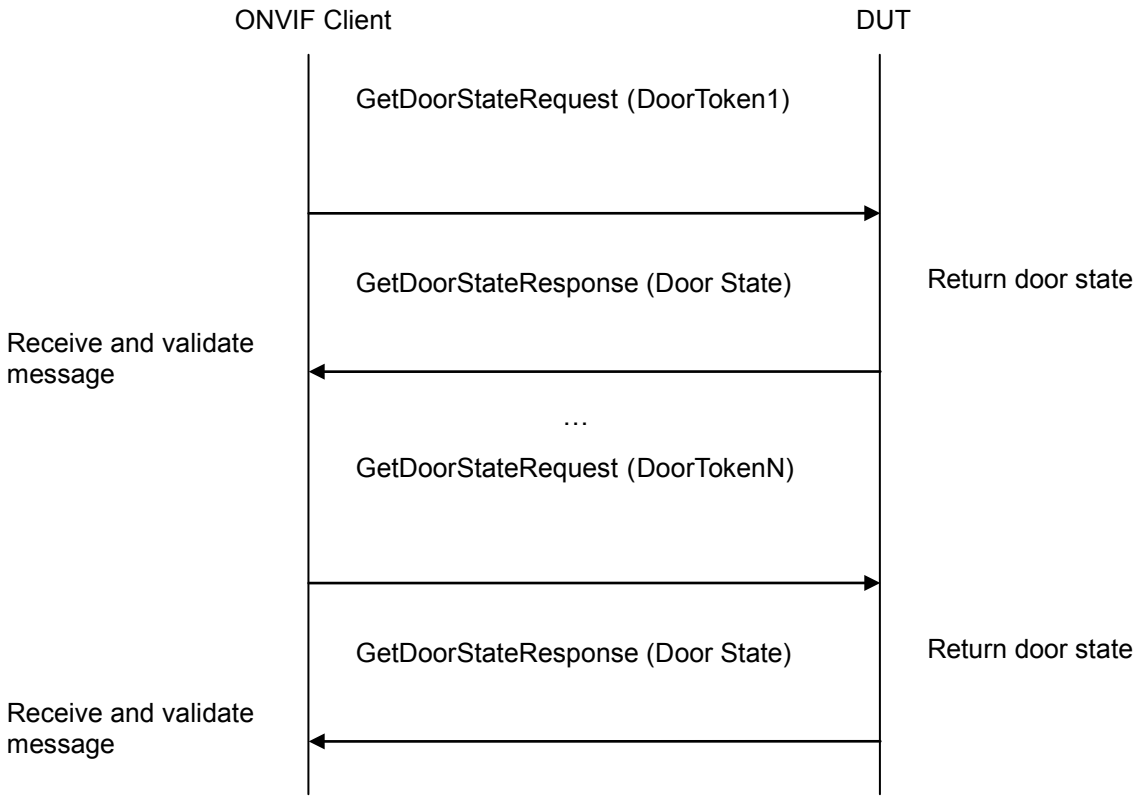
Test Purpose: To verify tns1:Door/State/DoorMode event generation after subscription and to verify tns1:Door/State/DoorMode event format.

Pre-Requisite: Event Service was received from the DUT. Door Control Service was received from the DUT. At least one Door 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 door list
Send complete door list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:Door/State/DoorMode topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/DoorMode”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
DUT generates events with initial state of DoorMode property
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorMode event messages for each Door)
Return messages
Receive and validate messages
…
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorMode event messages for each Door)
Return messages
Receive and validate messages



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors 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:Door/State/DoorMode. 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 a Property event (MessageDescription.IsProperty = "true").
8. Check that this event contains Source.SimpleItemDescription item with Name = "DoorToken" and Type = "pt:ReferenceToken".
9. Check that this event contains Data.SimpleItemDescription item with Name = «State» and Type = "tdc:DoorMode".
10. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.
11. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

12. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
13. Verify that the DUT sends a PullMessagesResponse that contains NotificationMessages. Repeat step 12 until Notification for all Doors is received.
14. Verify the received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
15. Verify that TopicExpression is equal to tns1:Door/State/DoorMode for all received Notify messages.
16. Verify that each notification contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to one of existing Door Tokens (e.g. complete list of doors contains Door with the same token). Verify that there are Notification messages for each Door.
17. Verify that each notification contains Data.SimpleItem item with Name = «State» and Value with type is equal to tdc:DoorMode.
18. Verify that Notify PropertyOperation = "Initialized".
19. ONVIF Client will invoke GetDoorStateRequest message for each Door with corresponding tokens.
20. Verify the GetDoorStateResponse messages from the DUT. Verify that Data.SimpleItem item with Name = «State» from Notification message has the same value with DoorMode elements from corresponding GetDoorStateResponse messages for each Door.

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:Door/State/DoorMode at least for one Door.

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

The DUT did not return at least one Door at step 3.

The DUT did not return Topic tns1:Door/State/DoorMode 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 13 will wait for Notification messages until notification for all Doors is received or Operation Delay has expired. Notification messages for all Doors are assumed as

received, if the number of Notification messages is equal to the number of Doors.

Note: If the DUT does not return any door for step 3, skip steps 4-23, 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 DOOR CONTROL – DOOR PHYSICAL STATE EVENT

Test Label: Door Control Service Property Events Check (Door Physical State).

Test Case ID: DOORCONTROL-6-1-2

ONVIF Core Specification Coverage: tns1:Door/State/DoorPhysicalState (ONVIF Door Control Service Specification), GetDoorState (ONVIF Door Control Service Specification), Properties (ONVIF Core Specification)

Command under test: GetDoorState

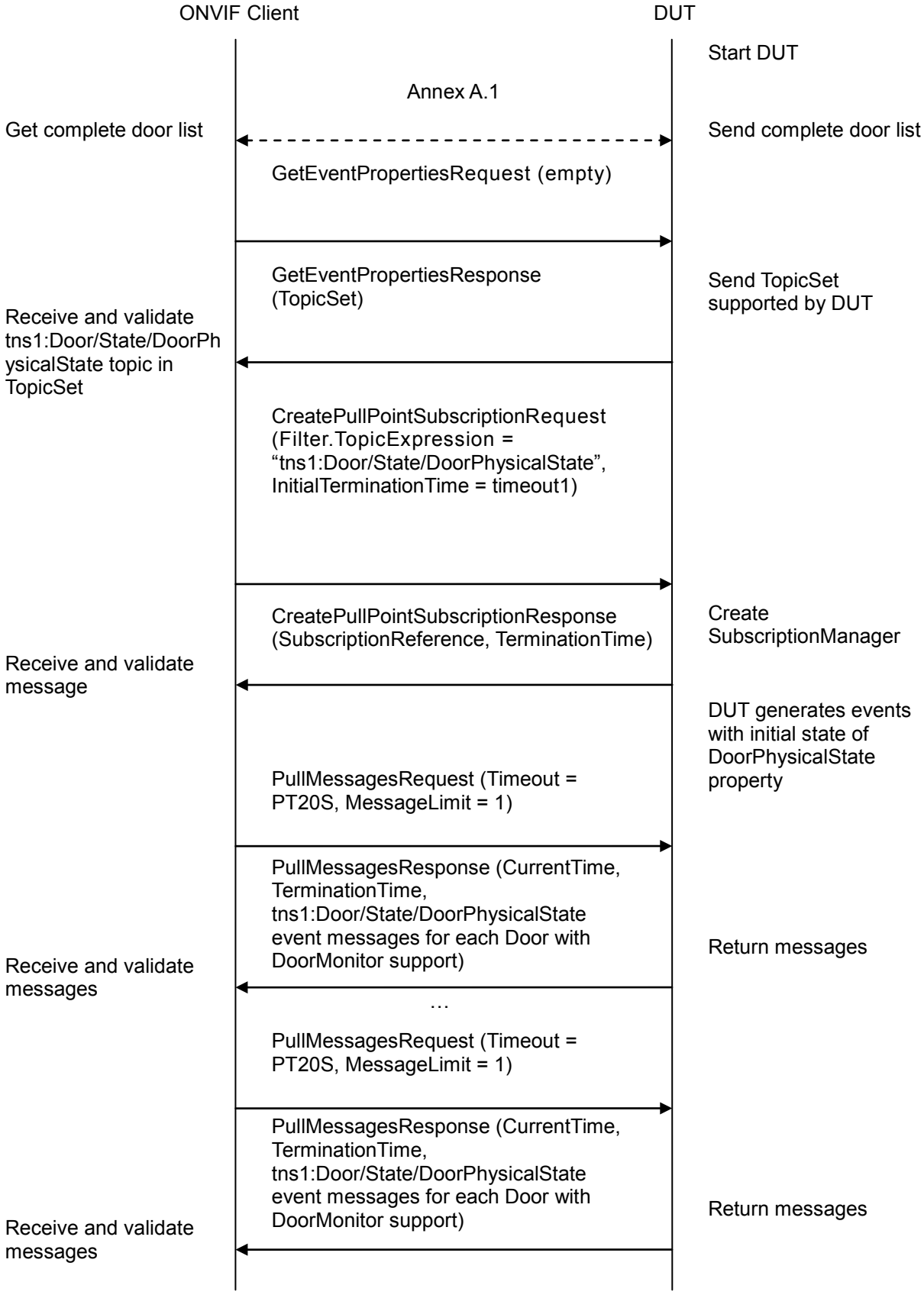
WSDL Reference: event.wsdl, doorcontrol.wsdl

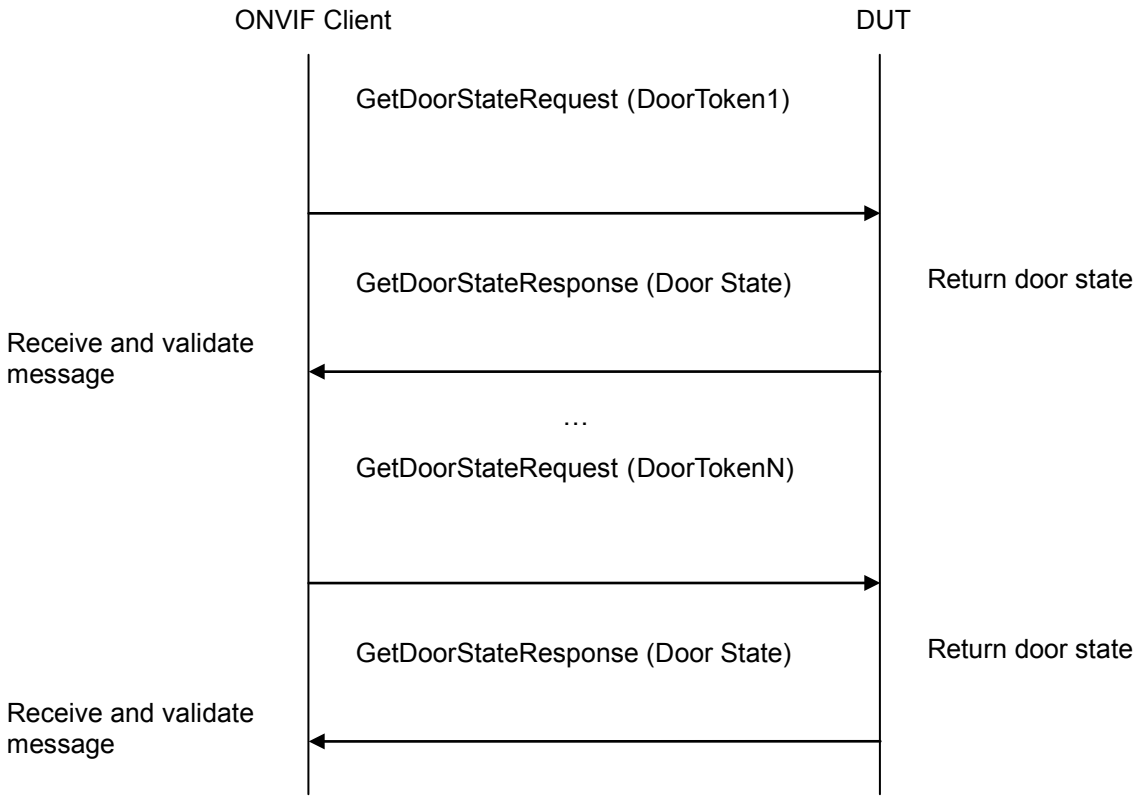
Test Purpose: To verify tns1:Door/State/DoorPhysicalState event generation after subscription and to verify tns1:Door/State/DoorPhysicalState event format.

Pre-Requisite: Event Service was received from the DUT. Door Control Service was received from the DUT. At least one Door with Door Monitor 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 door list
Send complete door list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:Door/State/DoorPhysicalState topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/DoorPhysicalState”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
DUT generates events with initial state of DoorPhysicalState property
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorPhysicalState event messages for each Door with DoorMonitor support)
Return messages
Receive and validate messages
…
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorPhysicalState event messages for each Door with DoorMonitor support)
Return messages
Receive and validate messages



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. Check that there is at least one Door with Capabilities.DoorMonitor = “true”. Otherwise 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:Door/State/DoorPhysicalState. 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 = "DoorToken" and Type = "pt:ReferenceToken".
10. Check that this event contains Data.SimpleItemDescription item with Name = "State" and Type = "tdc:DoorPhysicalState".
11. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorPhysicalState 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 Doors with Capabilities.DoorMonitor = “true” is received.
15. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
16. Verify that TopicExpression is equal to tns1:Door/State/DoorPhysicalState for all received Notify messages.
17. Verify that each notification contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to one of existing Door Tokens Capabilities.DoorMonitor = “true” (e.g. complete list of doors contains Door with the same token). Verify that there are Notification messages for each Door with Capabilities.DoorMonitor = “true”.
18. Verify that each notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to tdc:DoorPhysicalState.
19. Verify that Notify PropertyOperation = "Initialized".
20. ONVIF Client will invoke GetDoorStateRequest message for each Door with corresponding tokens.
21. Verify the GetDoorStateResponse messages from the DUT. Verify that Data.SimpleItem item with Name = "State" from Notification message has the same value with DoorPhysicalState elements from corresponding GetDoorStateResponse messages for each Door with Capabilities.DoorMonitor = “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 PullMessagesResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a Notification message that contains a property event tns1:Door/State/DoorPhysicalState at least for one Door with Capabilities.DoorMonitor = “true”.

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

The DUT did not return at least one Door at step 3.

The DUT did not return Topic tns1:Door/State/DoorPhysicalState 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 Doors with Capabilities.DoorMonitor = “true” is received or Operation Delay has expired. Notification messages for all Doors with Capabilities.DoorMonitor = “true” are assumed as received, if the number of Notification messages is equal to the number of Doors with Capabilities.DoorMonitor = “true”.

Note: If the DUT does not return any door for step 3, skip steps 4-23, 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.

4.5.3 DOOR CONTROL – DOOR PHYSICAL STATE EVENT STATE CHANGE

Test Label: Door Control Service Property Events State Change Check (Door Physical State).

Test Case ID: DOORCONTROL-6-1-3

ONVIF Core Specification Coverage: tns1:Door/State/DoorPhysicalState (ONVIF Door Control Service Specification)

Command under test: None

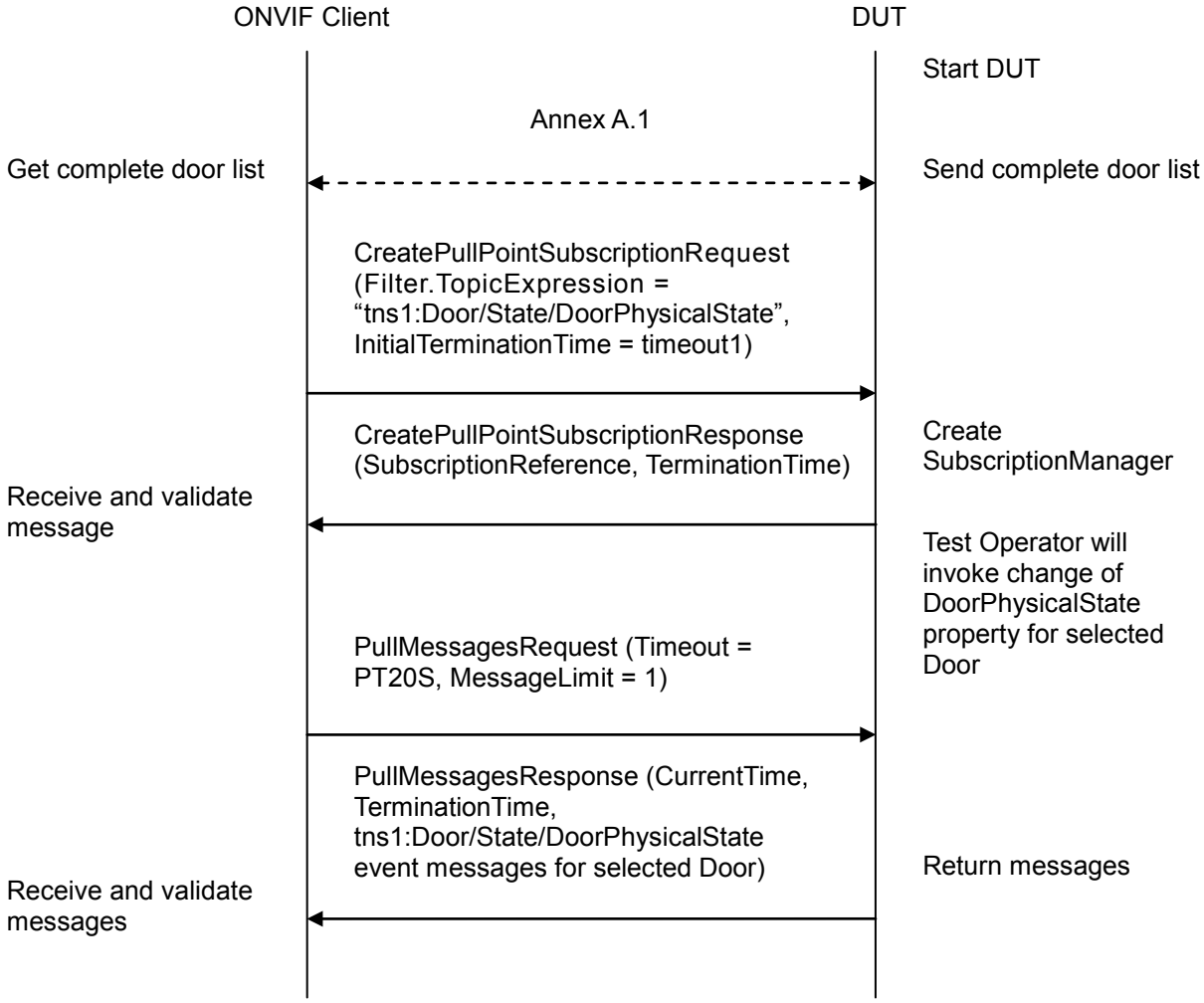
WSDL Reference: event.wsdl, doorcontrol.wsdl

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

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. Check that there is at least one Door with Capabilities.DoorMonitor = “true”. Otherwise fail the test and skip other steps.
5. ONVIF Client will select one first Door (token = **Token1**) with Capabilities.DoorMonitor = “true”.
6. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorPhysicalState Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
8. Test Operator will invoke change of DoorPhysicalState property for Door with token = **Token1**.

9. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
10. Verify that the DUT sends a PullMessagesResponse that contains NotificationMessages. Repeat step 9 until Notification with PropertyOperation = "Changed" for Door with Capabilities.DoorMonitor = “true” is received.
11. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
12. Verify that TopicExpression is equal to tns1:Door/State/DoorPhysicalState for received Notify message.
13. Verify that notification contains Source.SimpleItem item with Name = "DoorToken" and Value = “**Token1”**.
14. Verify that notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to tdc:DoorPhysicalState.

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:Door/State/DoorPhysicalState with valid DoorToken.

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

The DUT did not return at least one Door at step 3.

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 10 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

Note: If the DUT does not return any door for step 3, skip steps 4-14, 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.

4.5.4 DOOR CONTROL – DOUBLE LOCK PHYSICAL STATE EVENT

Test Label: Door Control Service Property Events Check (Double Lock Physical State).

Test Case ID: DOORCONTROL-6-1-4

ONVIF Core Specification Coverage: tns1:Door/State/DoubleLockPhysicalState (ONVIF Door Control Service Specification), GetDoorState (ONVIF Door Control Service Specification), Properties (ONVIF Core Specification)

Command under test: GetDoorState

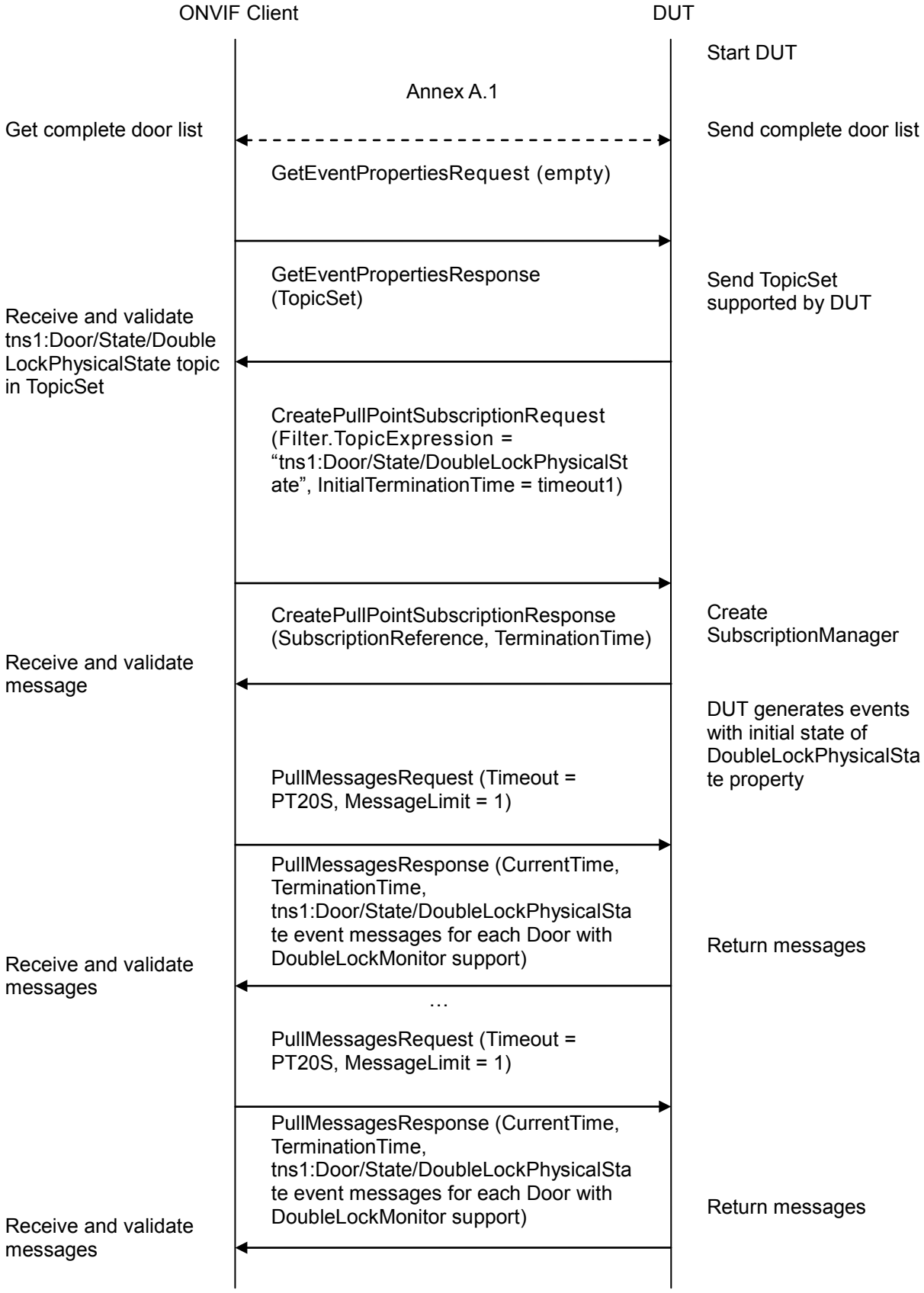
WSDL Reference: event.wsdl, doorcontrol.wsdl

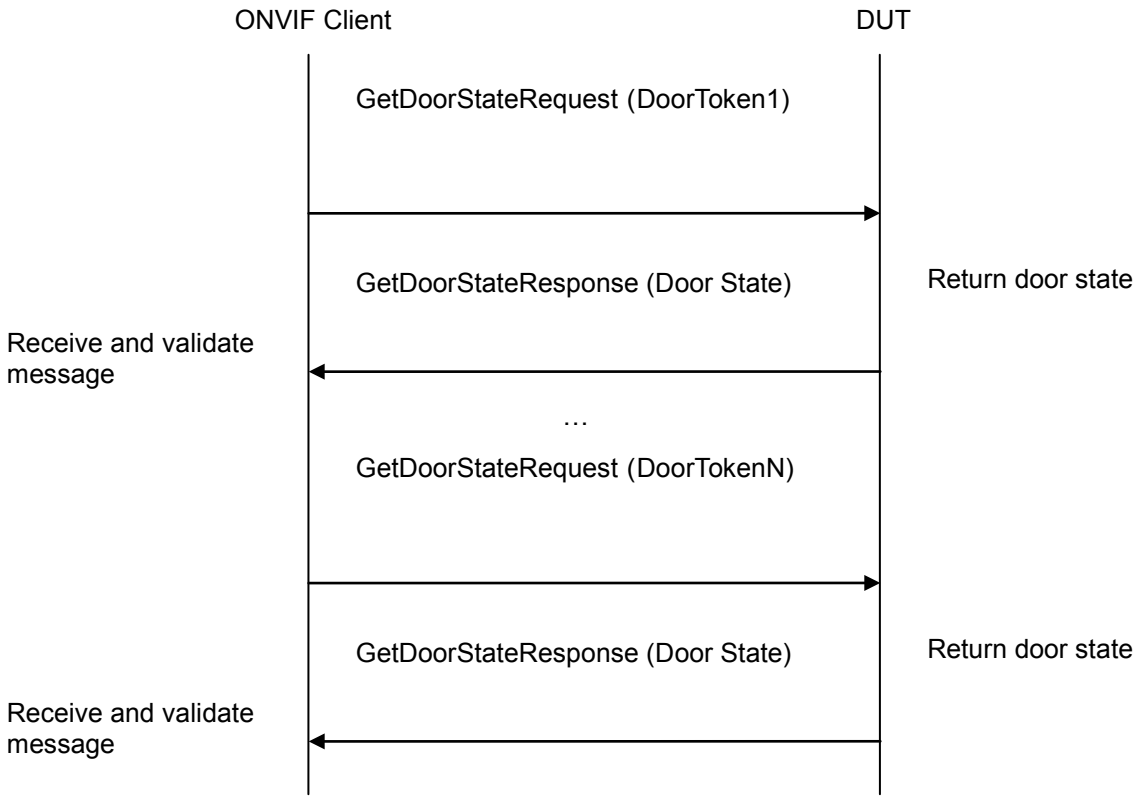
Test Purpose: To verify tns1:Door/State/DoubleLockPhysicalState event generation after subscription and to verify tns1:Door/State/DoubleLockPhysicalState event format.

Pre-Requisite: Event Service was received from the DUT. Door Control Service was received from the DUT. At least one Door with Double Lock Monitor 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 door list
Send complete door list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:Door/State/DoubleLockPhysicalState topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/DoubleLockPhysicalState”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
DUT generates events with initial state of DoubleLockPhysicalState property
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoubleLockPhysicalState event messages for each Door with DoubleLockMonitor support)
Return messages
Receive and validate messages
…
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoubleLockPhysicalState event messages for each Door with DoubleLockMonitor support)
Return messages
Receive and validate messages



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. Check that there is at least one Door with Capabilities.DoubleLockMonitor = “true”. Otherwise 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:Door/State/DoubleLockPhysicalState. 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 = "DoorToken" and Type = "pt:ReferenceToken".
10. Check that this event contains Data.SimpleItemDescription item with Name = "State" and Type = "tdc:LockPhysicalState".

11. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoubleLockPhysicalState 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 Doors with Capabilities.DoorMonitor = “true” is received.
15. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
16. Verify that TopicExpression is equal to tns1:Door/State/DoubleLockPhysicalState for all received Notify messages.
17. Verify that each notification contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to one of existing Door Tokens Capabilities.DoubleLockMonitor = “true” (e.g. complete list of doors contains Door with the same token). Verify that there are Notification messages for each Door with Capabilities.DoubleLockMonitor = “true”.
18. Verify that each notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to tdc:LockPhysicalState.
19. Verify that Notify PropertyOperation = "Initialized".
20. ONVIF Client will invoke GetDoorStateRequest message for each Door with corresponding tokens.
21. Verify the GetDoorStateResponse messages from the DUT. Verify that Data.SimpleItem item with Name = "State" from Notification message has the same value with DoubleLockPhysicalState elements from corresponding GetDoorStateResponse messages for each Door with Capabilities.DoubleLockMonitor = “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 PullMessagesResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a Notification message that contains a property event tns1:Door/State/DoubleLockPhysicalState at least for one Door with Capabilities.DoubleLockMonitor = “true”.

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

The DUT did not return at least one Door at step 3.

The DUT did not return Topic tns1:Door/State/DoubleLockPhysicalState 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 Doors with Capabilities.DoubleLockMonitor = “true” is received or Operation Delay has expired. Notification messages for all Doors with Capabilities.DoubleLockMonitor = “true” are assumed as received, if the number of Notification messages is equal to the number of Doors with Capabilities.DoubleLockMonitor = “true”.

Note: If the DUT does not return any door for step 3, skip steps 4-23, 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.

4.5.5 DOOR CONTROL – DOUBLE LOCK PHYSICAL STATE EVENT STATE CHANGE

Test Label: Door Control Service Property Events State Change Check (Double Lock Physical State).

Test Case ID: DOORCONTROL-6-1-5

ONVIF Core Specification Coverage: tns1:Door/State/DoubleLockPhysicalState (ONVIF Door Control Service Specification)

Command under test: None

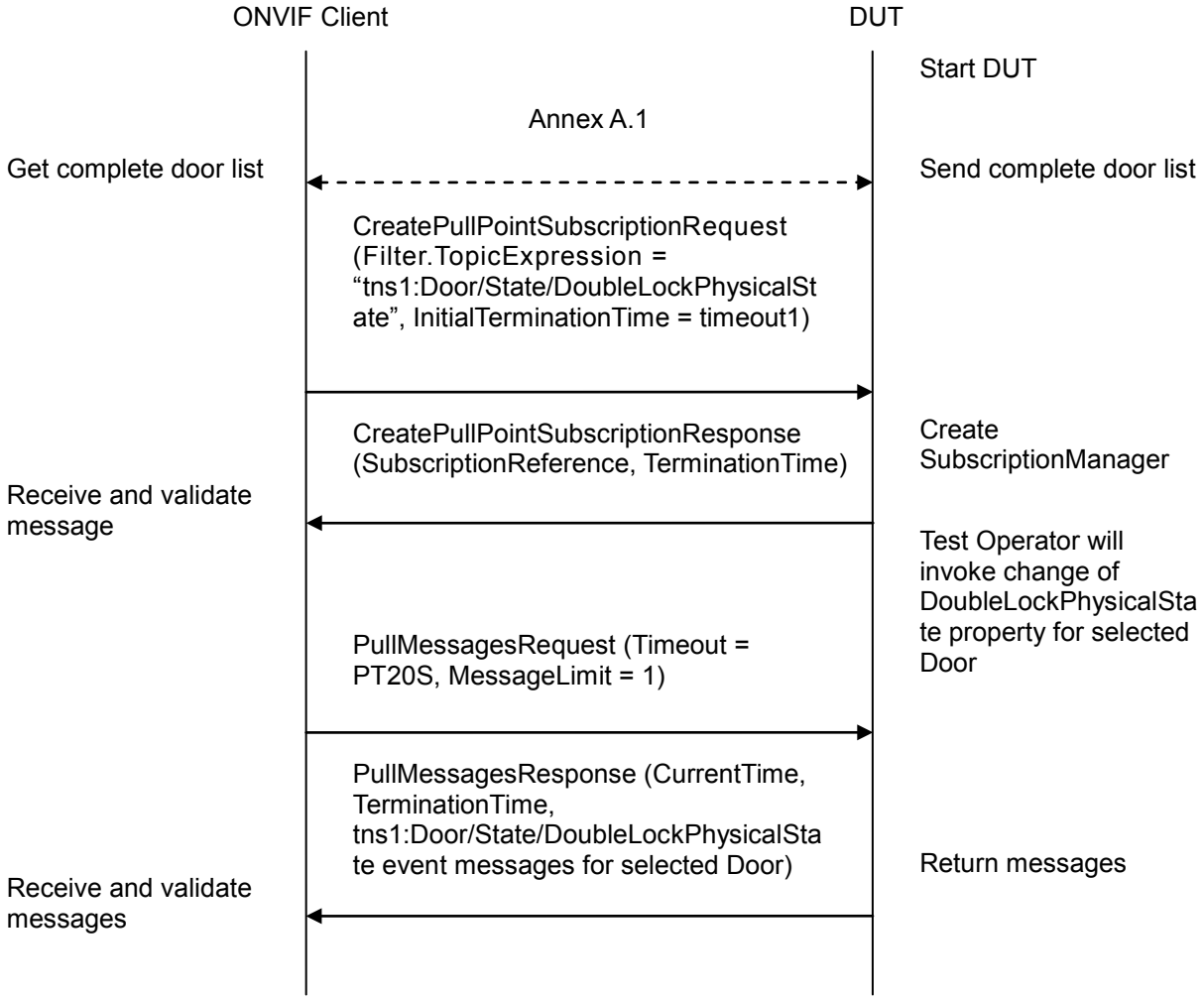
WSDL Reference: event.wsdl, doorcontrol.wsdl

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

Pre-Requisite: Event Service was received from the DUT. Door Control Service was received from the DUT. At least one Door with Double Lock Monitor capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. Check that there is at least one Door with Capabilities.DoubleLockMonitor = “true”. Otherwise fail the test and skip other steps.
5. ONVIF Client will select one first Door (token = **Token1**) with Capabilities.DoubleLockMonitor = “true”.
6. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoubleLockPhysicalState Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
8. Test Operator will invoke change of DoubleLockPhysicalState property for Door with token = **Token1**.

9. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
10. Verify that the DUT sends a PullMessagesResponse that contains NotificationMessages. Repeat step 9 until Notification with PropertyOperation = "Changed" for Door with Capabilities.DoubleLockMonitor = “true” is received.
11. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
12. Verify that TopicExpression is equal to tns1:Door/State/DoubleLockPhysicalState for received Notify message.
13. Verify that notification contains Source.SimpleItem item with Name = "DoorToken" and Value = “**Token1”**.
14. Verify that notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to tdc:DoubleLockPhysicalState.

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:Door/State/DoubleLockPhysicalState with valid DoorToken.

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

The DUT did not return at least one Door at step 3.

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 10 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

Note: If the DUT does not return any door for step 3, skip steps 4-14, 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.

4.5.6 DOOR CONTROL – LOCK PHYSICAL STATE EVENT

Test Label: Door Control Service Property Events Check (Lock Physical State).

Test Case ID: DOORCONTROL-6-1-6

ONVIF Core Specification Coverage: tns1:Door/State/LockPhysicalState (ONVIF Door Control Service Specification), GetDoorState (ONVIF Door Control Service Specification), Properties (ONVIF Core Specification)

Command under test: GetDoorState

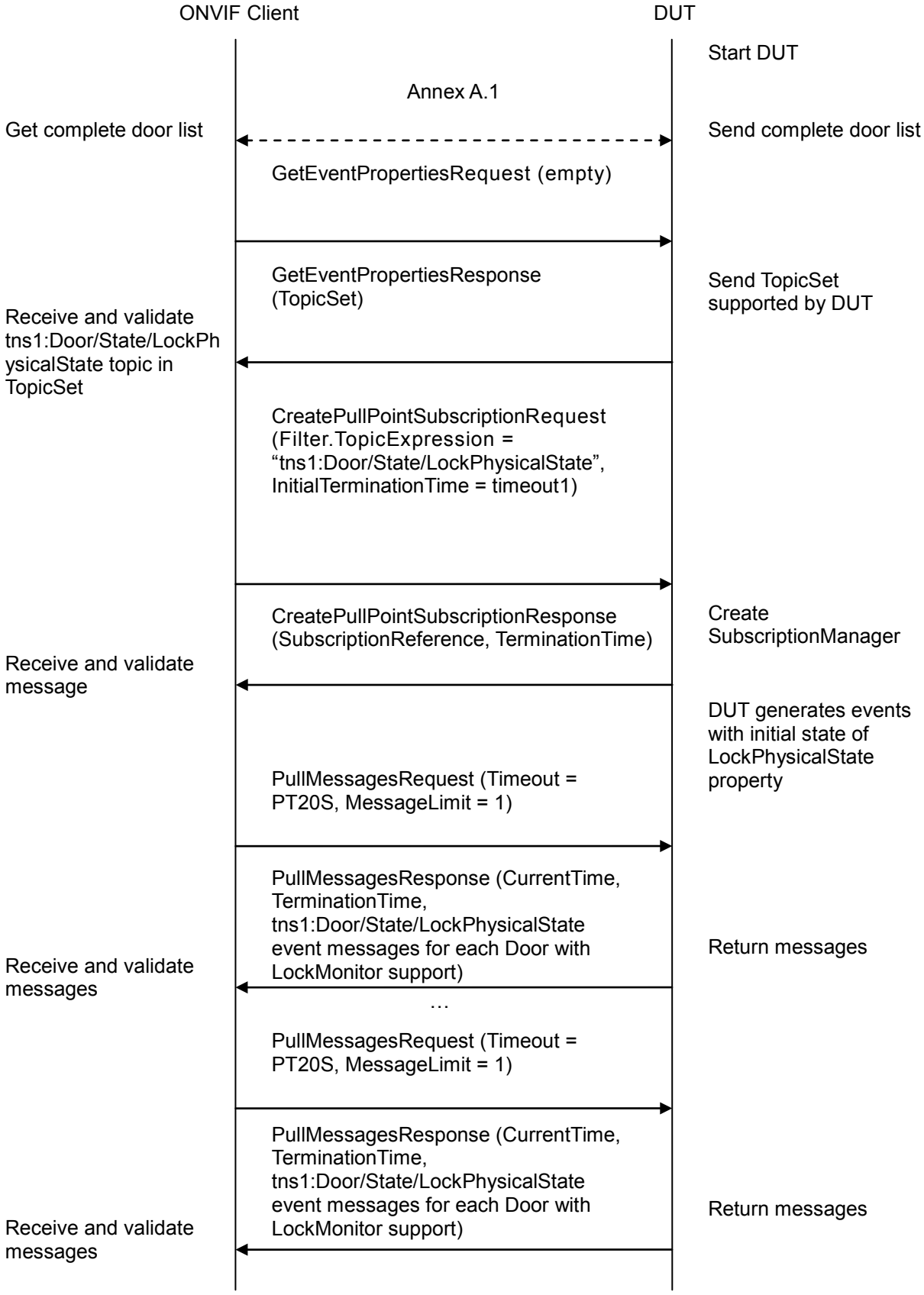
WSDL Reference: event.wsdl, doorcontrol.wsdl

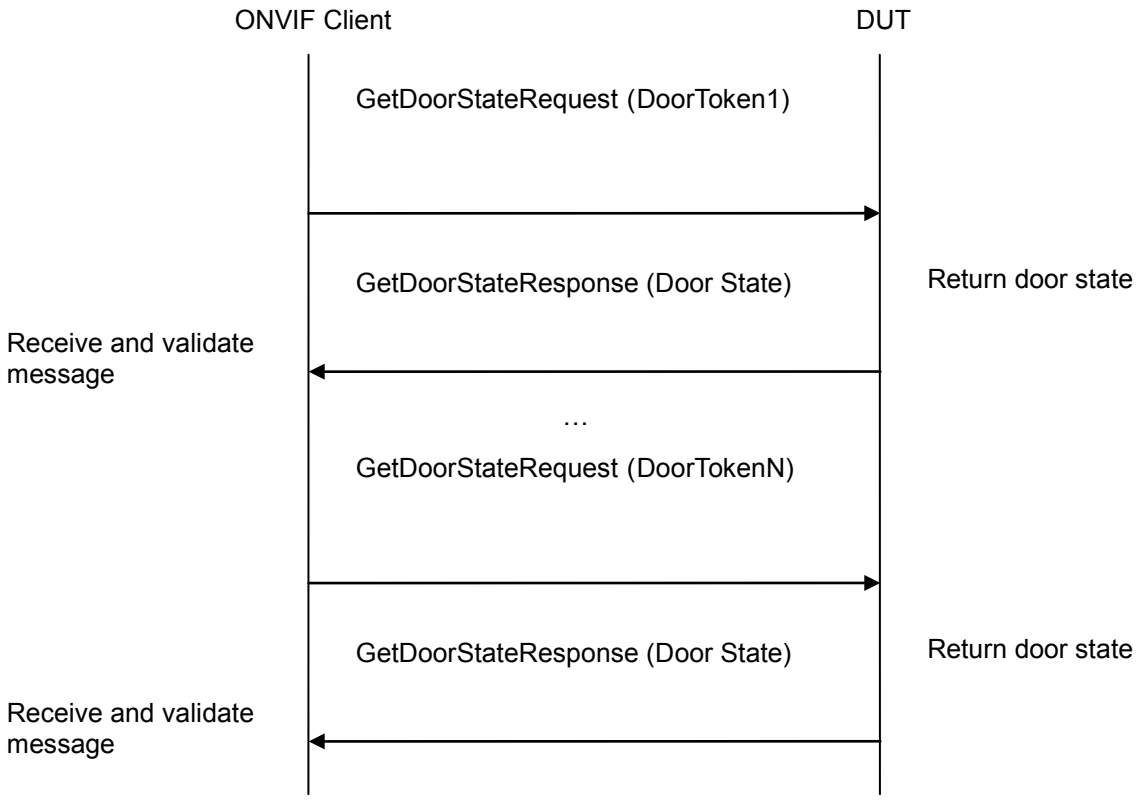
Test Purpose: To verify tns1:Door/State/LockPhysicalState event generation after subscription and to verify tns1:Door/State/LockPhysicalState event format.

Pre-Requisite: Event Service was received from the DUT. Door Control Service was received from the DUT. At least one Door with Lock Monitor 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 door list
Send complete door list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:Door/State/LockPhysicalState topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/LockPhysicalState”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
DUT generates events with initial state of LockPhysicalState property
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/LockPhysicalState event messages for each Door with LockMonitor support)
Return messages
Receive and validate messages
…
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/LockPhysicalState event messages for each Door with LockMonitor support)
Return messages
Receive and validate messages



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. Check that there is at least one Door with Capabilities.LockMonitor = “true”. Otherwise 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:Door/State/LockPhysicalState. 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 = "DoorToken" and Type = "pt:ReferenceToken".
10. Check that this event contains Data.SimpleItemDescription item with Name = "State" and Type = "tdc:LockPhysicalState".
11. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/LockPhysicalState 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 Doors with Capabilities.DoorMonitor = “true” is received.
15. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
16. Verify that TopicExpression is equal to tns1:Door/State/LockPhysicalState for all received Notify messages.
17. Verify that each notification contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to one of existing Door Tokens Capabilities.LockMonitor = “true” (e.g. complete list of doors contains Door with the same token). Verify that there are Notification messages for each Door with Capabilities.LockMonitor = “true”.
18. Verify that each notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to tdc:LockPhysicalState.
19. Verify that Notify PropertyOperation = "Initialized".
20. ONVIF Client will invoke GetDoorStateRequest message for each Door with corresponding tokens.
21. Verify the GetDoorStateResponse messages from the DUT. Verify that Data.SimpleItem item with Name = "State" from Notification message has the same value with LockPhysicalState elements from corresponding GetDoorStateResponse messages for each Door with Capabilities.LockMonitor = “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 PullMessagesResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a Notification message that contains a property event tns1:Door/State/LockPhysicalState at least for one Door with Capabilities.LockMonitor = “true”.

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

The DUT did not return at least one Door at step 3.

The DUT did not return Topic tns1:Door/State/LockPhysicalState 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 Doors with Capabilities.LockMonitor = “true” is received or Operation Delay has expired. Notification messages for all Doors with Capabilities.LockMonitor = “true” are assumed as received, if the number of Notification messages is equal to the number of Doors with Capabilities.LockMonitor = “true”.

Note: If the DUT does not return any door for step 3, skip steps 4-23, 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.

4.5.7 DOOR CONTROL – LOCK PHYSICAL STATE EVENT STATE CHANGE

Test Label: Door Control Service Property Events State Change Check (Lock Physical State).

Test Case ID: DOORCONTROL-6-1-7

ONVIF Core Specification Coverage: tns1:Door/State/LockPhysicalState (ONVIF Door Control Service Specification)

Command under test: None

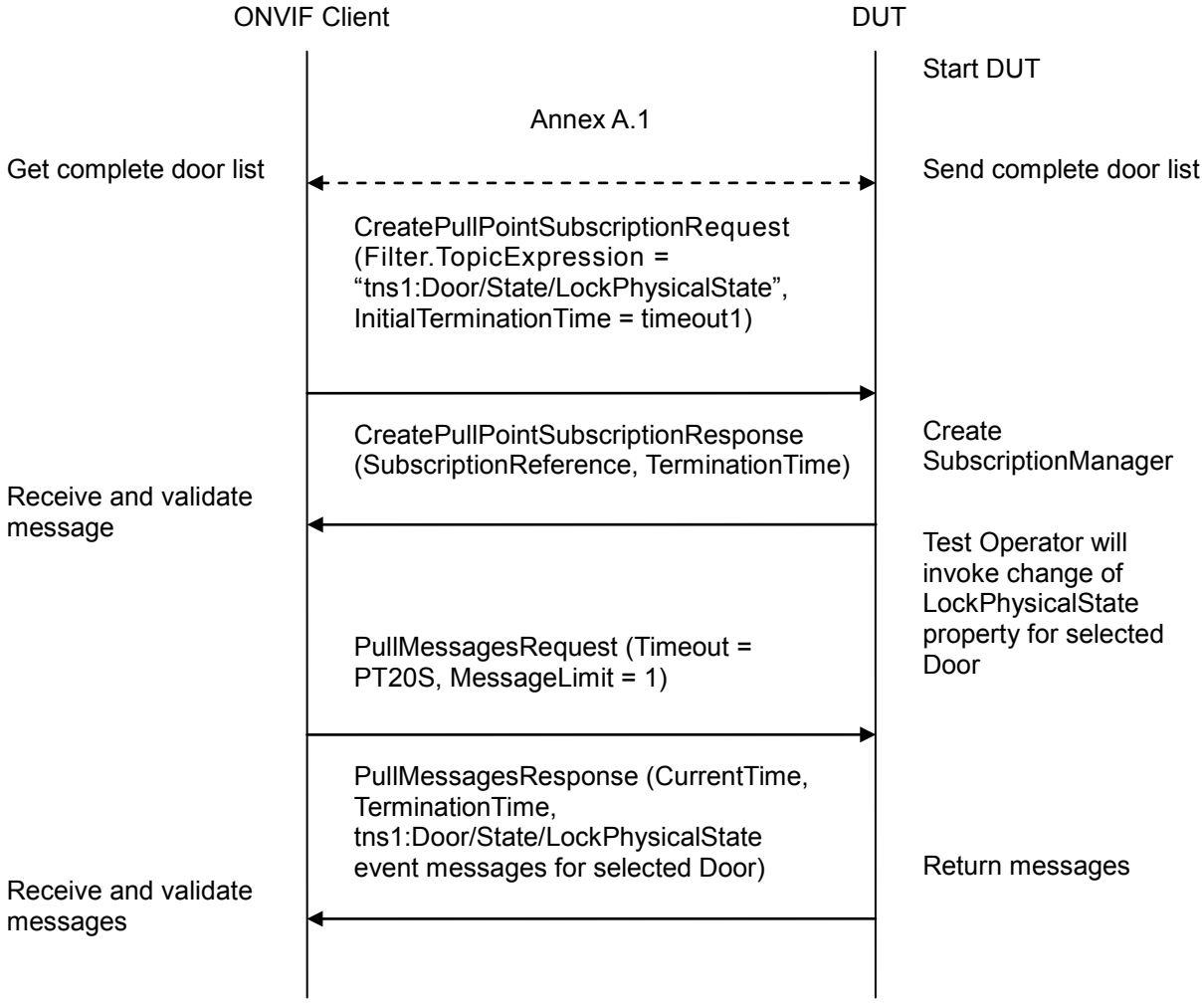
WSDL Reference: event.wsdl, doorcontrol.wsdl

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

Pre-Requisite: Event Service was received from the DUT. Door Control Service was received from the DUT. At least one Door with Lock Monitor capability is configured and added to the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. Check that there is at least one Door with Capabilities.LockMonitor = “true”. Otherwise, fail the test and skip other steps.
5. ONVIF Client will select one first Door (token = **Token1**) with Capabilities.LockMonitor = “true”.
6. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/LockPhysicalState Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
8. Test Operator will invoke change of LockPhysicalState property for Door with token = **Token1**.

9. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
10. Verify that the DUT sends a PullMessagesResponse that contains NotificationMessages. Repeat step 9 until Notification with PropertyOperation = "Changed" for Door with Capabilities.LockMonitor = “true” is received.
11. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
12. Verify that TopicExpression is equal to tns1:Door/State/LockPhysicalState for received Notify message.
13. Verify that notification contains Source.SimpleItem item with Name = "DoorToken" and Value = “**Token1”**.
14. Verify that notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to tdc:LockPhysicalState.

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:Door/State/LockPhysicalState with valid DoorToken.

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

The DUT did not return at least one Door at step 3.

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 10 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

Note: If the DUT does not return any door for step 3, skip steps 4-14, 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.

4.5.8 DOOR CONTROL – DOOR TAMPER EVENT

Test Label: Door Control Service Property Events Check (Door Tamper).

Test Case ID: DOORCONTROL-6-1-8

ONVIF Core Specification Coverage: tns1:Door/State/DoorTamper (ONVIF Door Control Service Specification), GetDoorState (ONVIF Door Control Service Specification), Properties (ONVIF Core Specification)

Command under test: GetDoorState

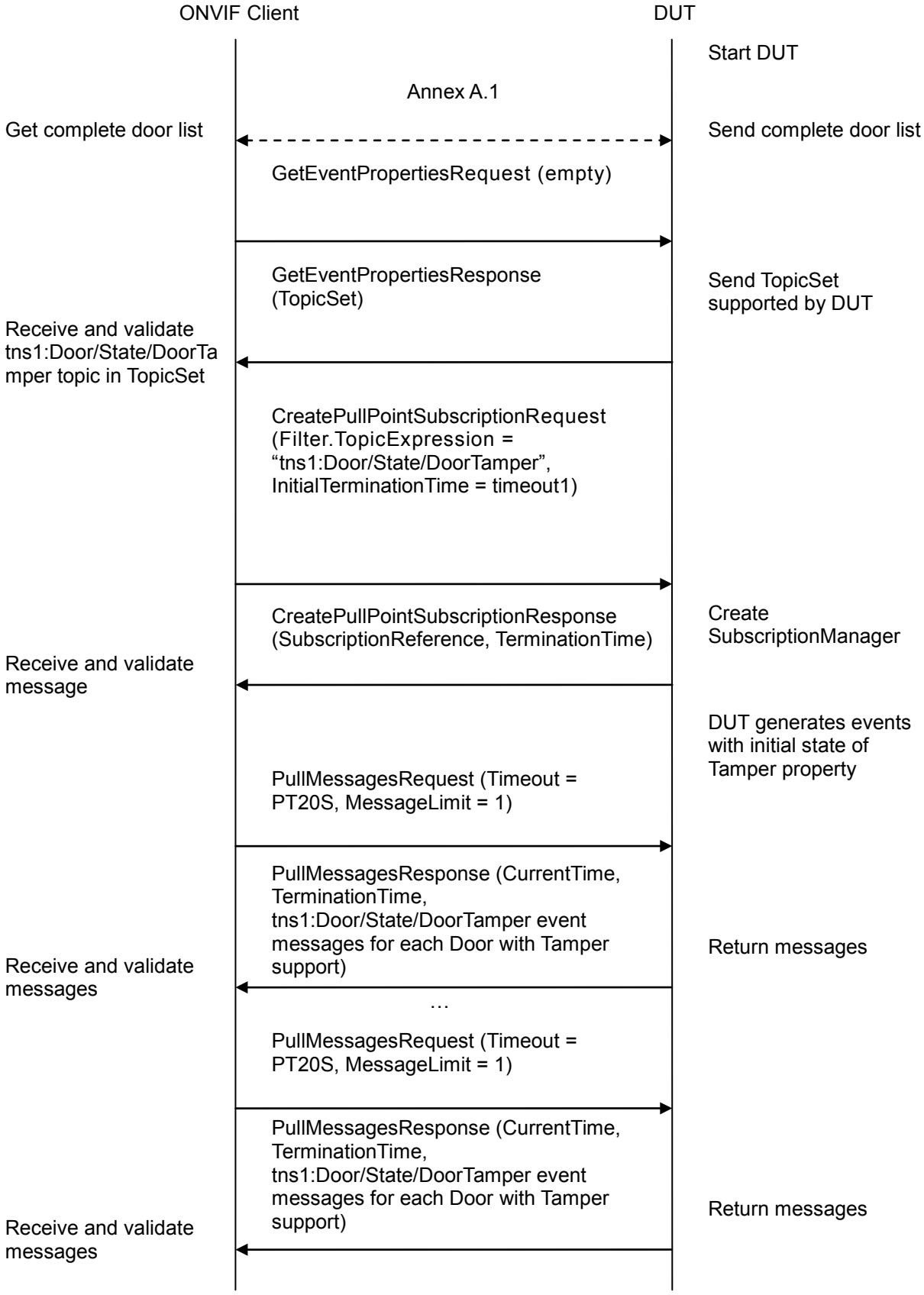
WSDL Reference: event.wsdl, doorcontrol.wsdl

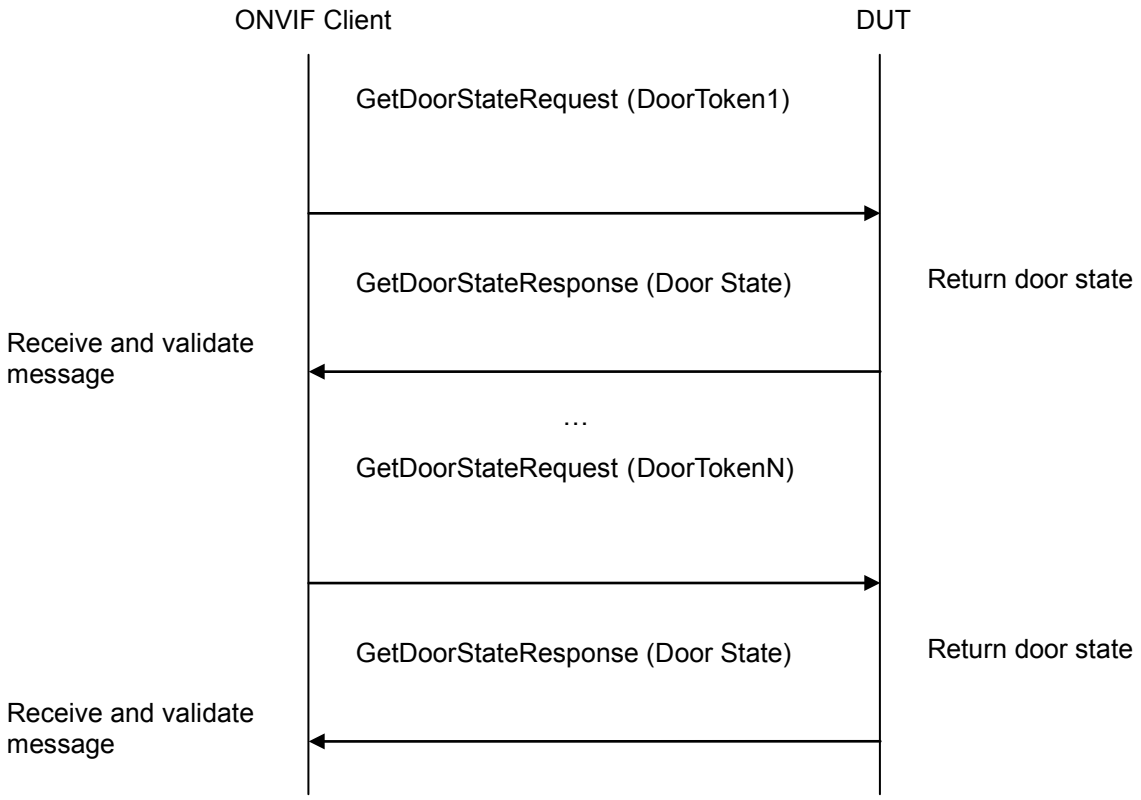
Test Purpose: To verify tns1:Door/State/DoorTamper event generation after subscription and to verify tns1:Door/State/DoorTamper event format.

Pre-Requisite: Event Service was received from the DUT. Door Control Service was received from the DUT. At least one Door with Tamper 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 door list
Send complete door list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:Door/State/DoorTamper topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/DoorTamper”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
DUT generates events with initial state of Tamper property
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorTamper event messages for each Door with Tamper support)
Return messages
Receive and validate messages
…
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorTamper event messages for each Door with Tamper support)
Return messages
Receive and validate messages



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. Check that there is at least one Door with Capabilities.Tamper = “true”. Otherwise 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:Door/State/DoorTamper. 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 = "DoorToken" and Type = "pt:ReferenceToken".
10. Check that this event contains Data.SimpleItemDescription item with Name = "State" and Type = "tdc:DoorTamperState".
11. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorTamper 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 Doors with Capabilities.DoorMonitor = “true” is received.
15. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
16. Verify that TopicExpression is equal to tns1:Door/State/DoorTamper for all received Notify messages.
17. Verify that each notification contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to one of existing Door Tokens Capabilities.Tamper = “true” (e.g. complete list of doors contains Door with the same token). Verify that there are Notification messages for each Door with Capabilities.Tamper = “true”.
18. Verify that each notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to tdc:DoorTamperState.
19. Verify that Notify PropertyOperation = "Initialized".
20. ONVIF Client will invoke GetDoorStateRequest message for each Door with corresponding tokens.
21. Verify the GetDoorStateResponse messages from the DUT. Verify that Data.SimpleItem item with Name = "State" from Notification message has the same value with Tamper elements from corresponding GetDoorStateResponse messages for each Door with Capabilities.Tamper = “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 PullMessagesResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a Notification message that contains a property event tns1:Door/State/DoorTamper at least for one Door with Capabilities.Tamper = “true”.

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

The DUT did not return at least one Door at step 3.

The DUT did not return Topic tns1:Door/State/DoorTamper 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 Doors with Capabilities.Tamper = “true” is received or Operation Delay has expired. Notification messages for all Doors with Capabilities.Tamper = “true” are assumed as received, if the number of Notification messages is equal to the number of Doors with Capabilities.Tamper = “true”.

Note: If the DUT does not return any door for step 3, skip steps 4-23, 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.

4.5.9 DOOR CONTROL – DOOR TAMPER EVENT STATE CHANGE

Test Label: Door Control Service Property Events State Change Check (Door Tamper).

Test Case ID: DOORCONTROL-6-1-9

ONVIF Core Specification Coverage: tns1:Door/State/DoorTamper (ONVIF Door Control Service Specification)

Command under test: None

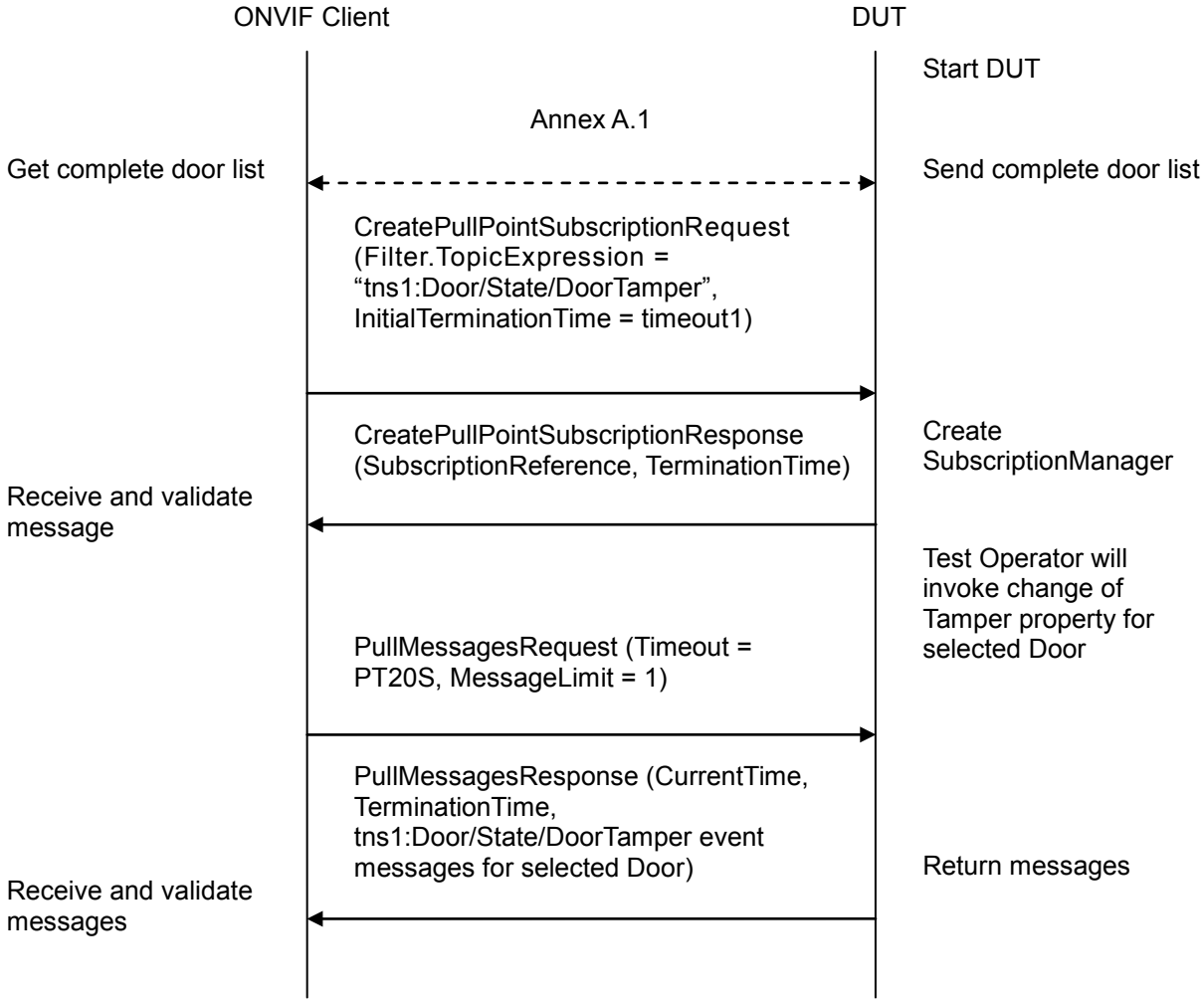
WSDL Reference: event.wsdl, doorcontrol.wsdl

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

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. Check that there is at least one Door with Capabilities.Tamper = “true”. Otherwise fail the test and skip other steps.
5. ONVIF Client will select one first Door (token = **Token1**) with Capabilities.Tamper = “true”.
6. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorTamper Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
8. Test Operator will invoke change of DoorTamper property for Door with token = **Token1**.
9. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

10. Verify that the DUT sends a PullMessagesResponse that contains NotificationMessages. Repeat step 9 until Notification with PropertyOperation = "Changed" for Door with Capabilities.Tamper = “true” is received.
11. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
12. Verify that TopicExpression is equal to tns1:Door/State/DoorTamper for received Notify message.
13. Verify that notification contains Source.SimpleItem item with Name = "DoorToken" and Value = “**Token1”**.
14. Verify that notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to tdc:DoorTamperState.

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:Door/State/DoorTamper with valid DoorToken.

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

The DUT did not return at least one Door at step 3.

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 10 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

Note: If the DUT does not return any door for step 3, skip steps 4-14, 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.

4.5.10 DOOR CONTROL – DOOR ALARM EVENT

Test Label: Door Control Service Property Events Check (Door Alarm).

Test Case ID: DOORCONTROL-6-1-10

ONVIF Core Specification Coverage: tns1:Door/State/DoorAlarm (ONVIF Door Control Service Specification), GetDoorState (ONVIF Door Control Service Specification), Properties (ONVIF Core Specification)

Command under test: GetDoorState

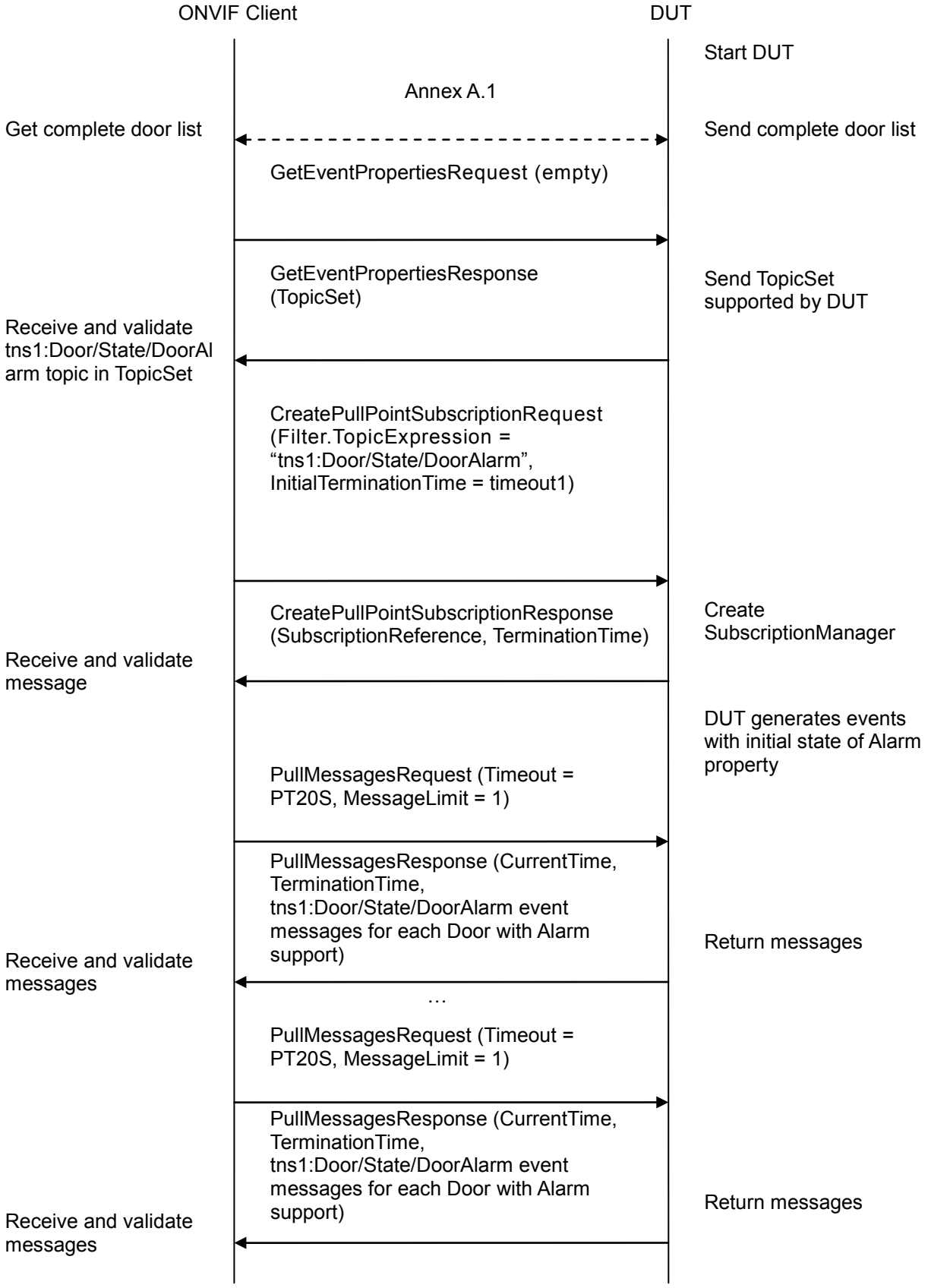
WSDL Reference: event.wsdl, doorcontrol.wsdl

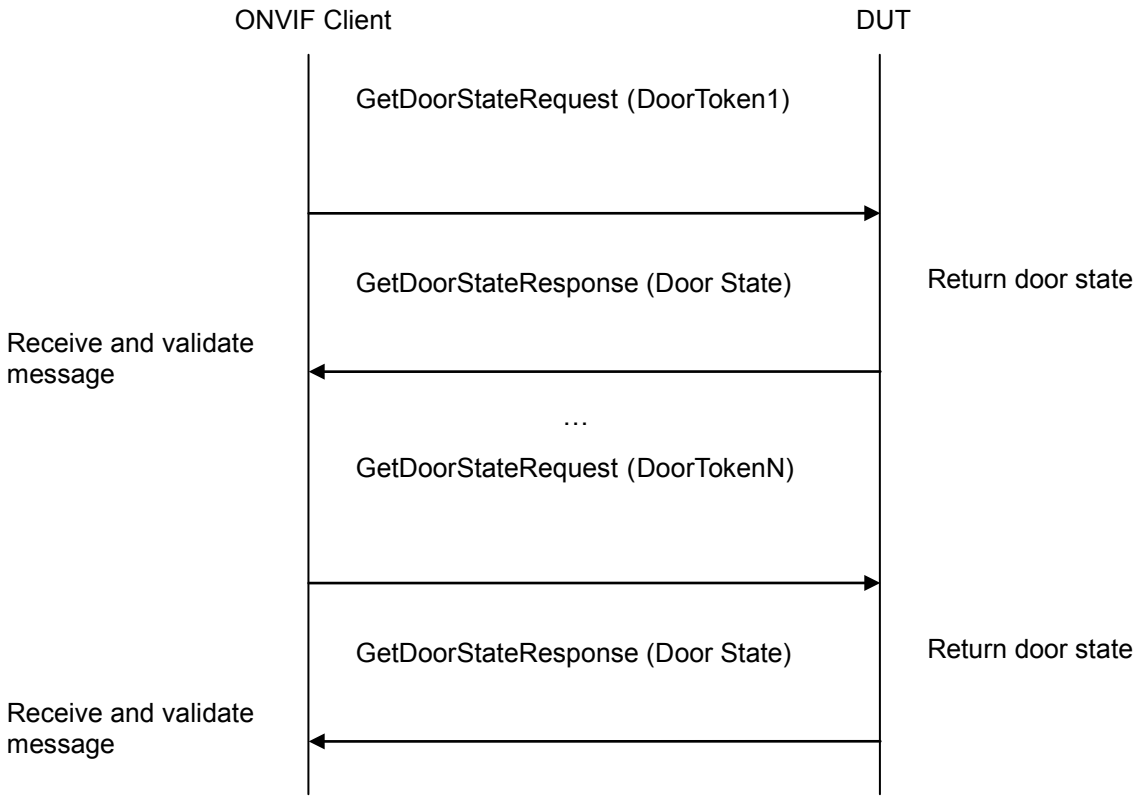
Test Purpose: To verify tns1:Door/State/DoorAlarm event generation after subscription and to verify tns1:Door/State/DoorAlarm event format.

Pre-Requisite: Event Service was received from the DUT. Door Control Service was received from the DUT. At least one Door with Alarm 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 door list
Send complete door list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:Door/State/DoorAlarm topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/DoorAlarm”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
DUT generates events with initial state of Alarm property
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorAlarm event messages for each Door with Alarm support)
Return messages
Receive and validate messages
…
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorAlarm event messages for each Door with Alarm support)
Return messages
Receive and validate messages



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. Check that there is at least one Door with Capabilities.Alarm = “true”. Otherwise 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:Door/State/DoorAlarm. 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 = "DoorToken" and Type = "pt:ReferenceToken".
10. Check that this event contains Data.SimpleItemDescription item with Name = "State" and Type = "tdc:DoorAlarmState".
11. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorAlarm 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 Doors with Capabilities.DoorMonitor = “true” is received.
15. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
16. Verify that TopicExpression is equal to tns1:Door/State/DoorAlarm for all received Notify messages.
17. Verify that each notification contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to one of existing Door Tokens Capabilities.Alarm = “true” (e.g. complete list of doors contains Door with the same token). Verify that there are Notification messages for each Door with Capabilities.Alarm = “true”.
18. Verify that each notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to tdc:DoorAlarmState.
19. Verify that Notify PropertyOperation = "Initialized".
20. ONVIF Client will invoke GetDoorStateRequest message for each Door with corresponding tokens.
21. Verify the GetDoorStateResponse messages from the DUT. Verify that Data.SimpleItem item with Name = "State" from Notification message has the same value with Alarm elements from corresponding GetDoorStateResponse messages for each Door with Capabilities.Alarm = “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 PullMessagesResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a Notification message that contains a property event tns1:Door/State/DoorAlarm at least for one Door with Capabilities.Alarm = “true”.

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

The DUT did not return at least one Door at step 3.

The DUT did not return Topic tns1:Door/State/DoorAlarm 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 Doors with Capabilities.Alarm = “true” is received or Operation Delay has expired. Notification messages for all Doors with Capabilities.Alarm = “true” are assumed as received, if the number of Notification messages is equal to the number of Doors with Capabilities.Alarm = “true”.

Note: If the DUT does not return any door for step 3, skip steps 4-23, 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.

4.5.11 DOOR CONTROL – DOOR ALARM EVENT STATE CHANGE

Test Label: Door Control Service Property Events State Change Check (Door Alarm).

Test Case ID: DOORCONTROL-6-1-11

ONVIF Core Specification Coverage: tns1:Door/State/DoorAlarm (ONVIF Door Control Service Specification)

Command under test: None

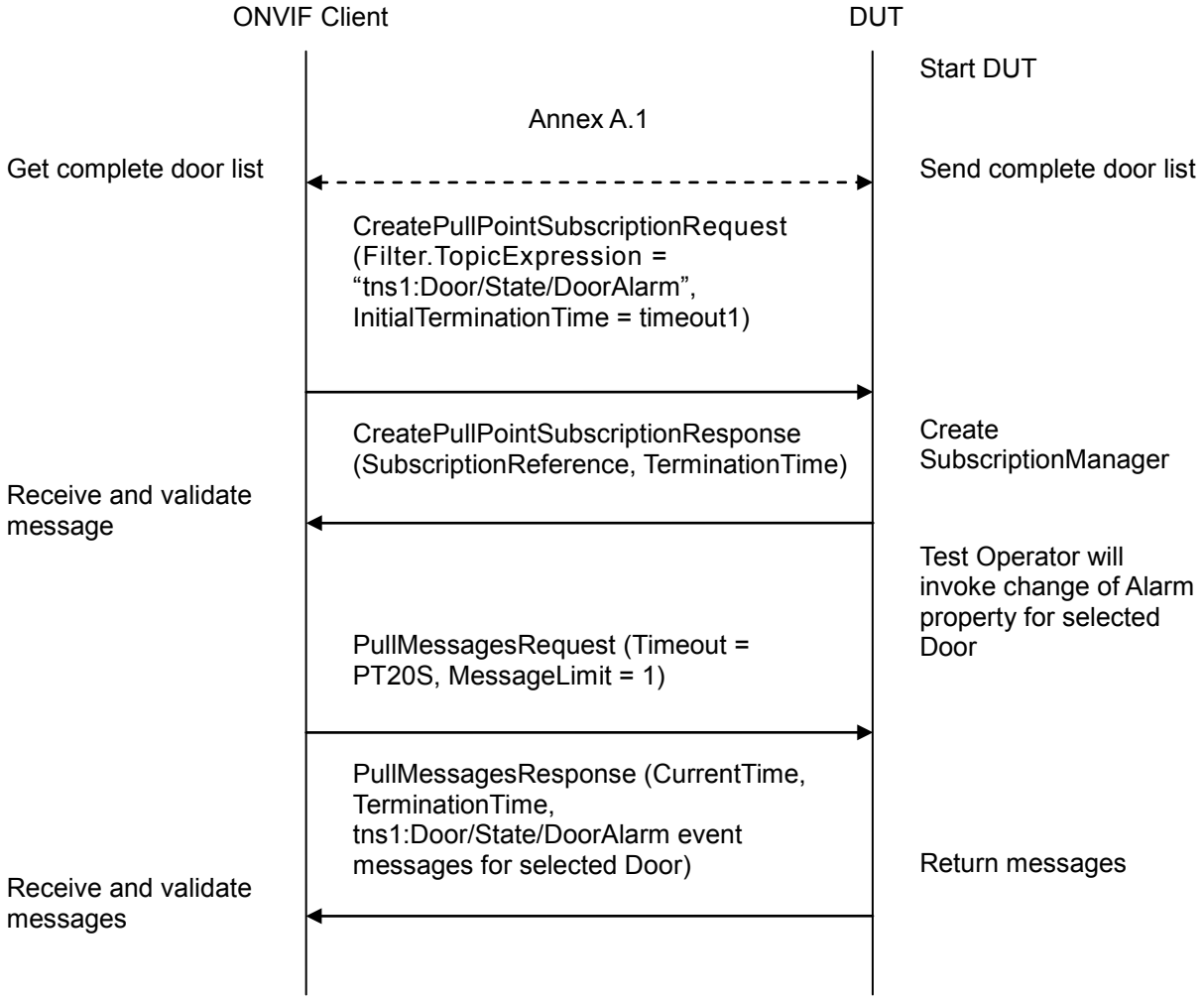
WSDL Reference: event.wsdl, doorcontrol.wsdl

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

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. Check that there is at least one Door with Capabilities.Alarm = “true”. Otherwise fail the test and skip other steps.
5. ONVIF Client will select one first Door (token = **Token1**) with Capabilities.Alarm = “true”.
6. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorAlarm Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
8. Test Operator will invoke change of DoorAlarm property for Door with token = **Token1**.
9. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

10. Verify that the DUT sends a PullMessagesResponse that contains NotificationMessages. Repeat step 9 until Notification with PropertyOperation = "Changed" for Door with Capabilities.Alarm = “true” is received.
11. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
12. Verify that TopicExpression is equal to tns1:Door/State/DoorAlarm for received Notify message.
13. Verify that notification contains Source.SimpleItem item with Name = "DoorToken" and Value = “**Token1”**.
14. Verify that notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to tdc:DoorAlarmState.

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:Door/State/DoorAlarm with valid DoorToken.

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

The DUT did not return at least one Door at step 3.

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 10 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

Note: If the DUT does not return any door for step 3, skip steps 4-14, 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.

4.5.12 DOOR CONTROL – DOOR FAULT EVENT

Test Label: Door Control Service Property Events Check (Door Fault).

Test Case ID: DOORCONTROL-6-1-12

ONVIF Core Specification Coverage: tns1:Door/State/DoorFault (ONVIF Door Control Service Specification), GetDoorState (ONVIF Door Control Service Specification), Properties (ONVIF Core Specification)

Command under test: GetDoorState

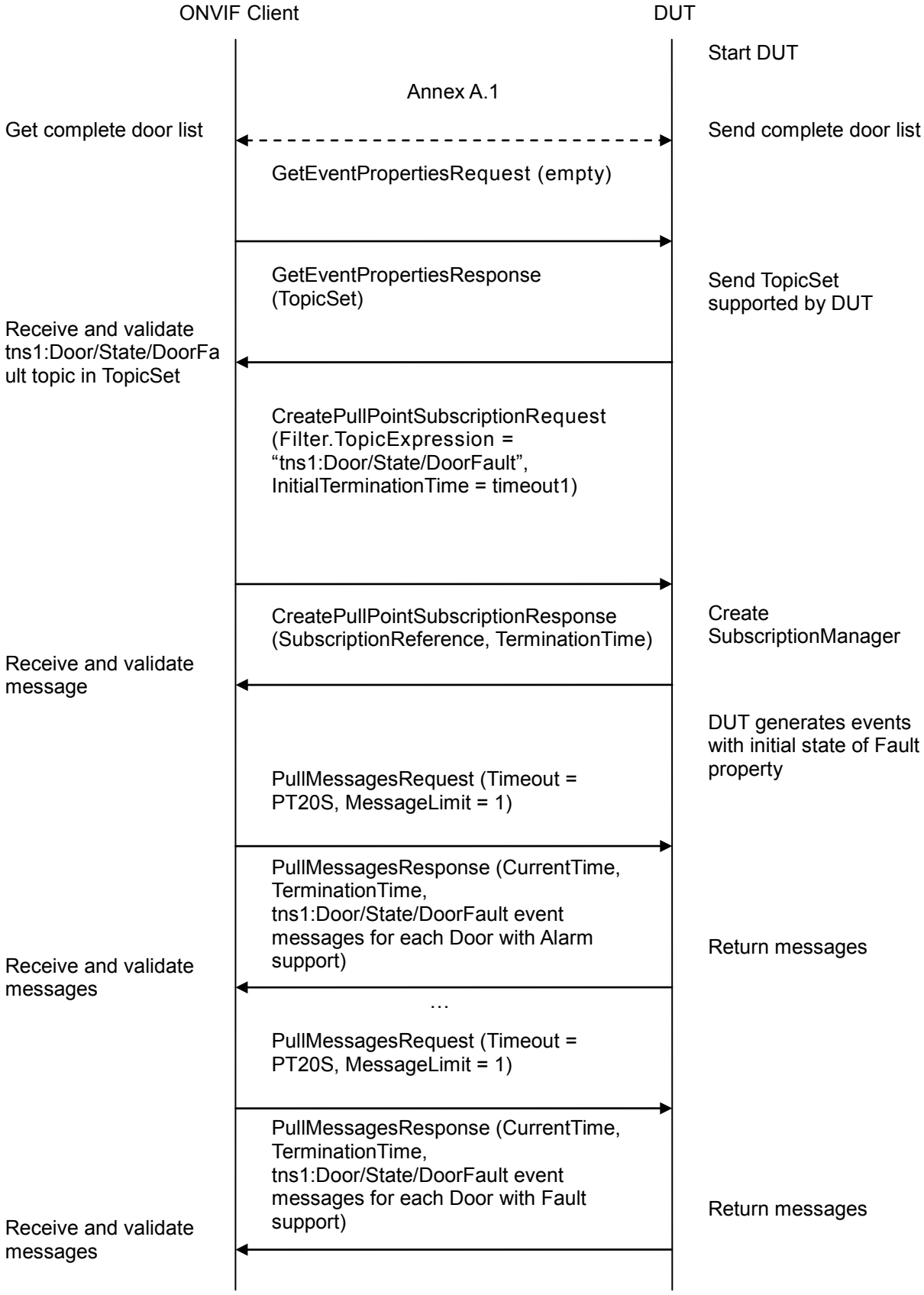
WSDL Reference: event.wsdl, doorcontrol.wsdl

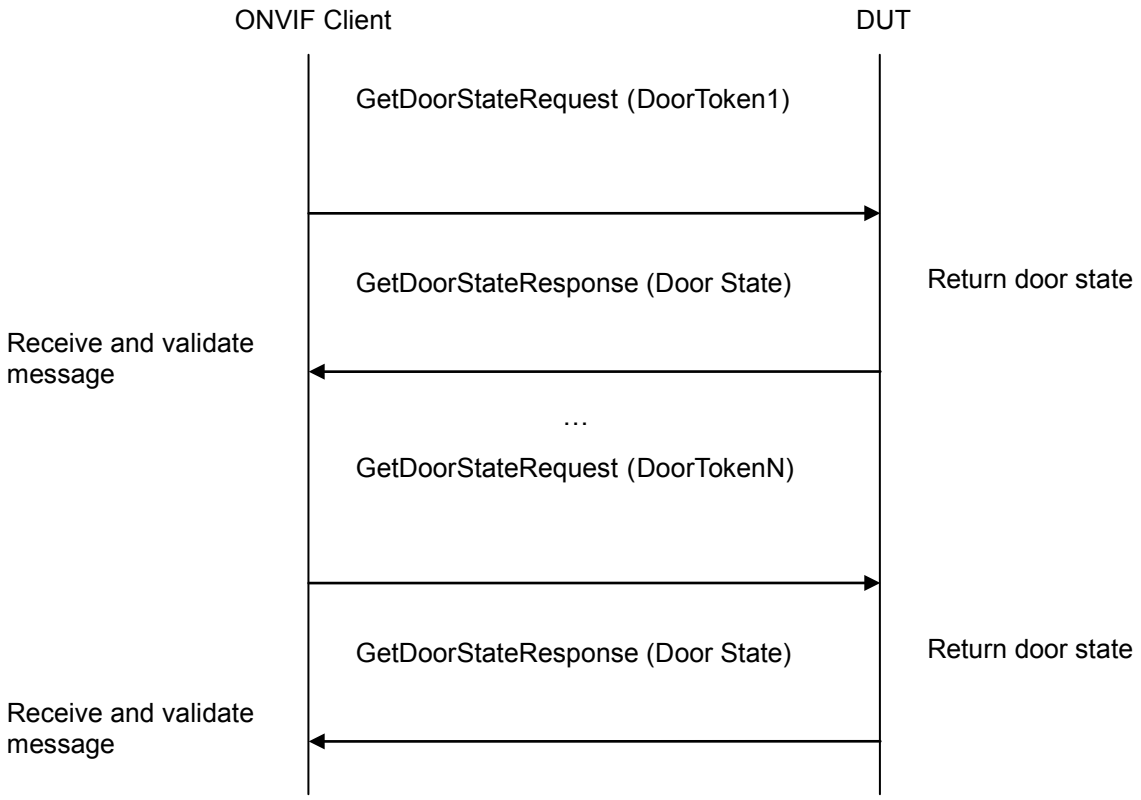
Test Purpose: To verify tns1:Door/State/DoorFault event generation after subscription and to verify tns1:Door/State/DoorFault event format.

Pre-Requisite: Event Service was received from the DUT. Door Control Service was received from the DUT. At least one Door with Fault 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 door list
Send complete door list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:Door/State/DoorFault topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Door/State/DoorFault”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
DUT generates events with initial state of Fault property
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorFault event messages for each Door with Alarm support)
Return messages
Receive and validate messages
…
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Door/State/DoorFault event messages for each Door with Fault support)
Return messages
Receive and validate messages



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. Check that there is at least one Door with Capabilities.Fault = “true”. Otherwise 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:Door/State/DoorFault. 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 = "DoorToken" and Type = "pt:ReferenceToken".
10. Check that this event contains Data.SimpleItemDescription item with Name = "State" and Type = "tdc:DoorFaultState".
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:Door/State/DoorFault Topic as Filter and an InitialTerminationTime of timeout1.
13. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
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 NotificationMessages. Repeat step 14 until Notification for all Doors with Capabilities.DoorMonitor = “true” is received.
16. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
17. Verify that TopicExpression is equal to tns1:Door/State/DoorFault for all received Notify messages.
18. Verify that each notification contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to one of existing Door Tokens Capabilities.Fault = “true” (e.g. complete list of doors contains Door with the same token). Verify that there are Notification messages for each Door with Capabilities.Fault = “true”.
19. Verify that each notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to tdc:DoorFaultState.
20. Verify that each notification which contains Data.SimpleItem item with Name = "Reason" has Value with type is equal to xs:string.
21. Verify that Notify PropertyOperation = "Initialized".
22. ONVIF Client will invoke GetDoorStateRequest message for each Door with corresponding tokens.
23. Verify the GetDoorStateResponse messages from the DUT. Verify that Data.SimpleItem item with Name = "State" from Notification message has the same value with Fault elements from corresponding GetDoorStateResponse messages for each Door with Capabilities.Fault = “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 PullMessagesResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a Notification message that contains a property event tns1:Door/State/DoorFault at least for one Door with Capabilities.Fault = “true”.

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

The DUT did not return at least one Door at step 3.

The DUT did not return Topic tns1:Door/State/DoorFault 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 15 will wait for Notification messages until notification for all Doors with Capabilities.Fault = “true” is received or Operation Delay has expired. Notification messages for all Doors with Capabilities.Fault = “true” are assumed as received, if the number of Notification messages is equal to the number of Doors with Capabilities.Fault = “true”.

Note: If the DUT does not return any door for step 3, skip steps 4-23, 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.

4.5.13 DOOR CONTROL – DOOR FAULT EVENT STATE CHANGE

Test Label: Door Control Service Property Events State Change Check (Door Fault).

Test Case ID: DOORCONTROL-6-1-13

ONVIF Core Specification Coverage: tns1:Door/State/DoorFault (ONVIF Door Control Service Specification)

Command under test: None

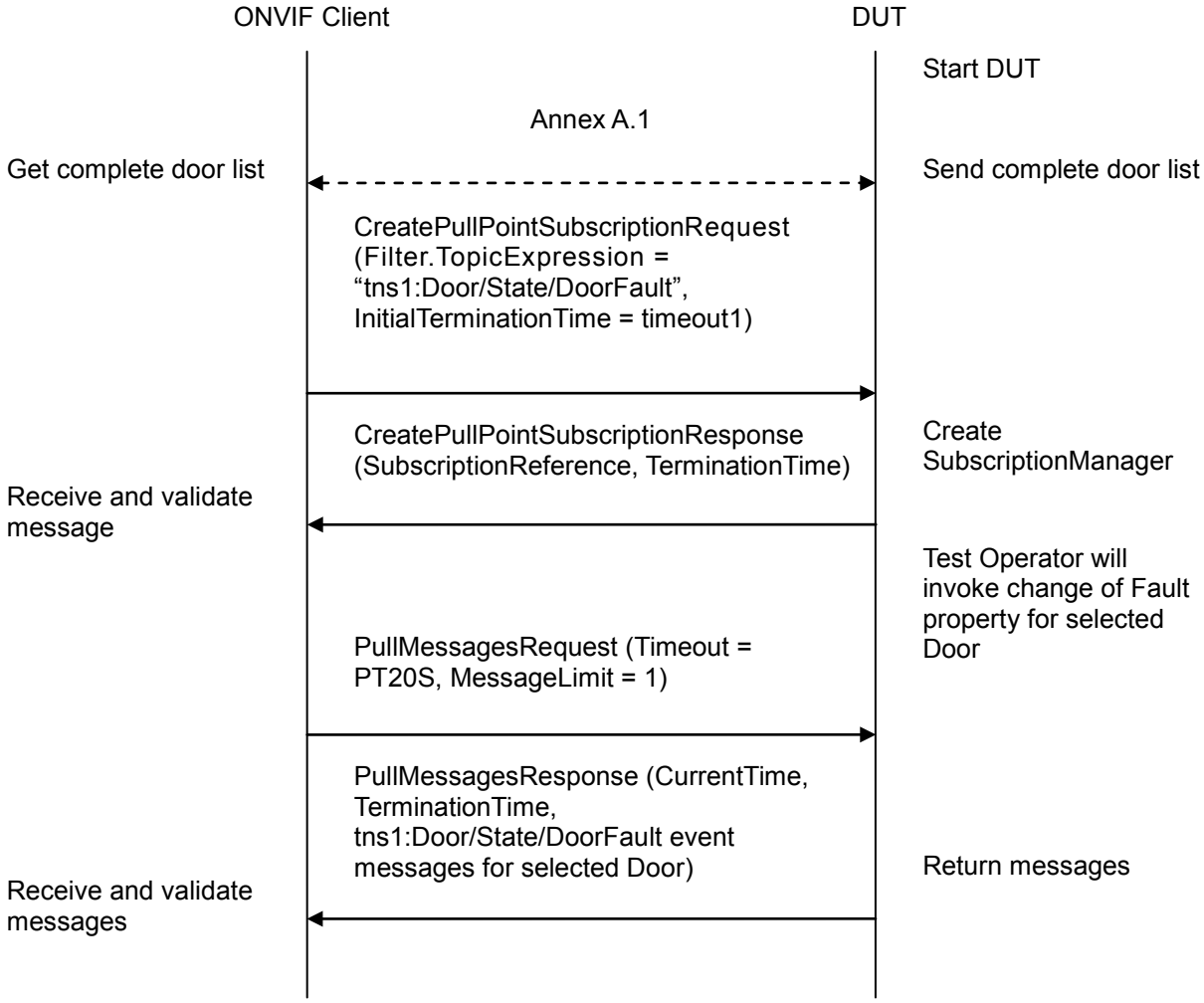
WSDL Reference: event.wsdl, doorcontrol.wsdl

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

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. Get a complete list of doors from the DUT (see Annex A.1).
4. Check that there is at least one Door with Capabilities.Fault = “true”. Otherwise, fail the test and skip other steps.
5. ONVIF Client will select one first Door (token = **Token1**) with Capabilities.Fault = “true”.
6. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorFault Topic as Filter and an InitialTerminationTime of timeout1.
7. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
8. Test Operator will invoke change of DoorFault property for Door with token = **Token1**.
9. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.

10. Verify that the DUT sends a PullMessagesResponse that contains NotificationMessages. Repeat step 9 until Notification with PropertyOperation = "Changed" for Door with Capabilities.Fault = “true” is received.
11. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
12. Verify that TopicExpression is equal to tns1:Door/State/DoorFault for received Notify message.
13. Verify that notification contains Source.SimpleItem item with Name = "DoorToken" and Value = “**Token1”**.
14. Verify that notification contains Data.SimpleItem item with Name = "State" and Value with type is equal to tdc:DoorFaultState.
15. Verify that notification contains Data.SimpleItem item with Name = "Reason" and Value with type is equal to xs:string.

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:Door/State/DoorFault with valid DoorToken.

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

The DUT did not return at least one Door at step 3.

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 10 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

Note: If the DUT does not return any door for step 3, skip steps 4-15, 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.

4.6 Door Configuration

4.6.1 DOOR CONTROL – ADD OR CHANGE DOOR EVENT

Test Label: Door Control Service Add or Change Door Event Check.

Test Case ID: DOORCONTROL-7-1-1

ONVIF Core Specification Coverage: tns1:Configuration/Door/Changed (ONVIF Door Control Service Specification), GetDoorInfo (ONVIF Door Control Service Specification)

Command under test: GetDoorInfo

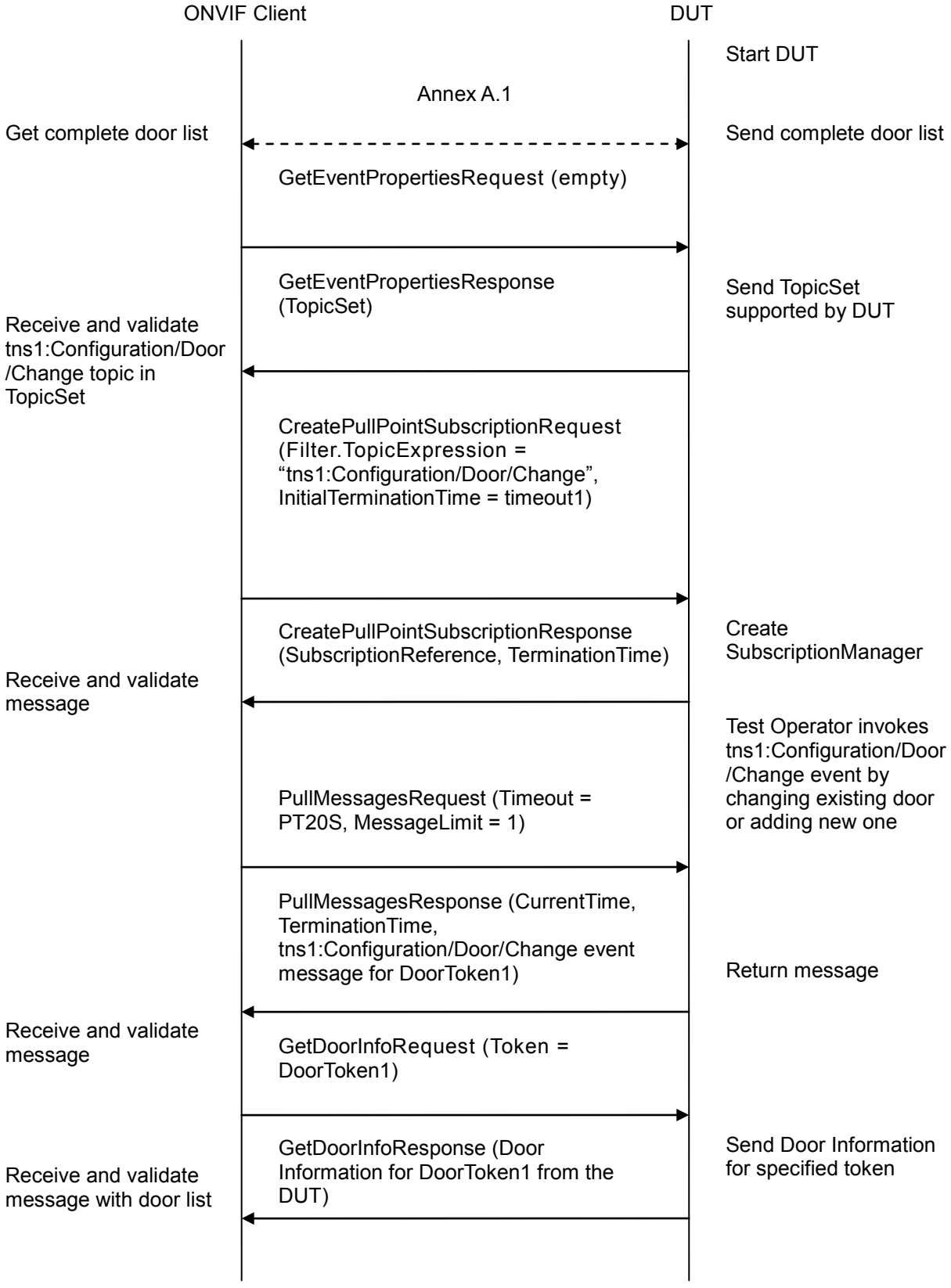
WSDL Reference: event.wsdl, doorcontrol.wsdl

Test Purpose: To verify tns1:Configuration/Door/Changed event generation after adding new door or changing door configuration to the DUT and to verify tns1:Configuration/Door/Changed event format.

Pre-Requisite: Event Service was received from the DUT. Door Control Service was received from the DUT. At least one Door could be added to the DUT or changed.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.1
Get complete door list
Send complete door list
GetEventPropertiesRequest (empty)
GetEventPropertiesResponse (TopicSet)
Send TopicSet supported by DUT
Receive and validate tns1:Configuration/Door /Change topic in TopicSet
CreatePullPointSubscriptionRequest (Filter.TopicExpression = “tns1:Configuration/Door/Change”, InitialTerminationTime = timeout1)
CreatePullPointSubscriptionResponse (SubscriptionReference, TerminationTime)
Create SubscriptionManager
Receive and validate message
Test Operator invokes tns1:Configuration/Door /Change event by changing existing door or adding new one
PullMessagesRequest (Timeout = PT20S, MessageLimit = 1)
PullMessagesResponse (CurrentTime, TerminationTime, tns1:Configuration/Door/Change event message for DoorToken1)
Return message
Receive and validate message
GetDoorInfoRequest (Token = DoorToken1)
GetDoorInfoResponse (Door Information for DoorToken1 from the DUT)
Send Door Information for specified token
Receive and validate message with door 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/Door/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 = "DoorToken" and Type = "pt:ReferenceToken".
8. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Configuration/Door/Changed Topic as Filter and an InitialTerminationTime of timeout1.
9. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
10. Test Operator will add Door or change Door 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 NotificationMessages. Repeat step 11 until Notification received.
13. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
14. Verify that TopicExpression is equal to tns1:Configuration/Door/Changed for received message.
15. Verify that notification contains Source.SimpleItem item with Name = "DoorToken" and Value with type is equal to tdc:ReferenceToken.
16. ONVIF Client will invoke GetDoorInfoRequest message (**Token** from Notify message) to retrieve a subset of Door Information from the DUT.
17. Verify the GetDoorInfoResponse message from the DUT. Verify that requested Door 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 GetDoorInfoResponse with specified Door.

The DUT did not send a Notification message that contains an event tns1:Configuration/Door/Changed with valid DoorToken.

The DUT sent an invalid Notification message (invalid DoorToken value).

The DUT did not return at least one Door at step 3.

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 12 will wait for 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.

4.6.2 DOOR CONTROL – REMOVE DOOR EVENT

Test Label: Door Control Service Remove Door Event Check.

Test Case ID: DOORCONTROL-7-1-2

ONVIF Core Specification Coverage: tns1:Configuration/Door/Removed (ONVIF Door Control Service Specification), GetDoorInfo (ONVIF Door Control Service Specification)

Command under test: GetDoorInfo

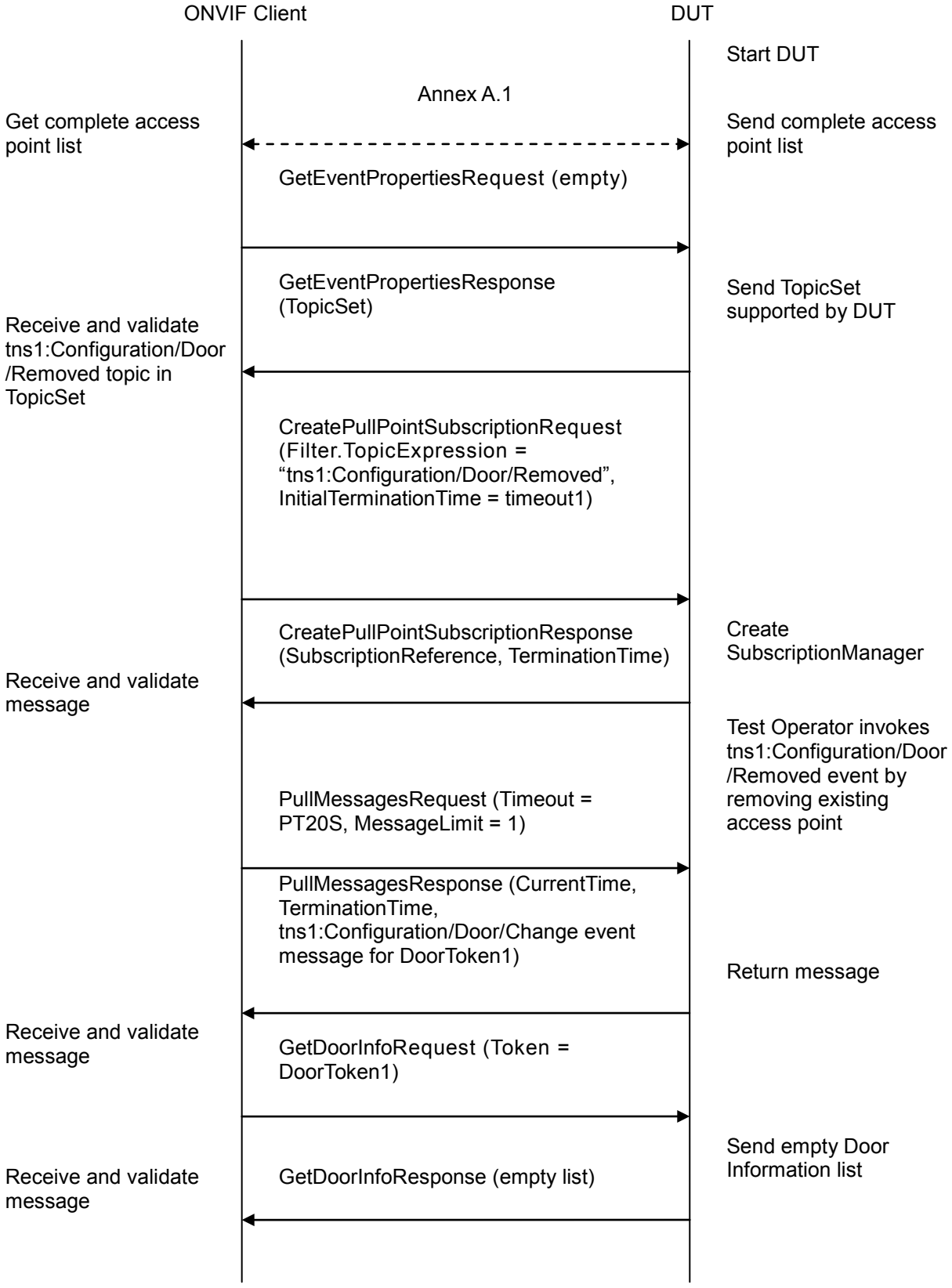
WSDL Reference: event.wsdl, doorcontrol.wsdl

Test Purpose: To verify tns1: Configuration/Door/Removed event generation after removing door from the DUT and to verify tns1: Configuration/Door/Removed event format.

Pre-Requisite: Event Service was received from the DUT. Door Control Service was received from the DUT. At least one Door could be removed from the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:



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/Door/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 = "DoorToken" and Type = "pt:ReferenceToken".
8. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Configuration/Door/Removed Topic as Filter and an InitialTerminationTime of timeout1.
9. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
10. Test Operator will remove Door.
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 NotificationMessages. Repeat step 11 until Notification received.
13. Verify received Notify messages (correct value for UTC time, TopicExpression and wsnt:Message).
14. Verify that TopicExpression is equal to tns1:Configuration/Door/Removed for received message.
15. Verify that notification contains Source.SimpleItem item with Name = "DoorToken" and Value with type is equal to tdc:ReferenceToken.
16. ONVIF Client will invoke GetDoorInfoRequest message (**Token** from Notify message) to retrieve subset of Door Information from the DUT.
17. Verify the GetDoorInfoResponse 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 PullMessagesResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a GetDoorInfoResponse with empty list.

The DUT did not send a Notification message that contains an event tns1:Configuration/Door/Removed with valid DoorToken.

The DUT sent an invalid Notification message (invalid DoorToken value).

The DUT did not return at least one Door at step 3.

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 12 will wait for 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

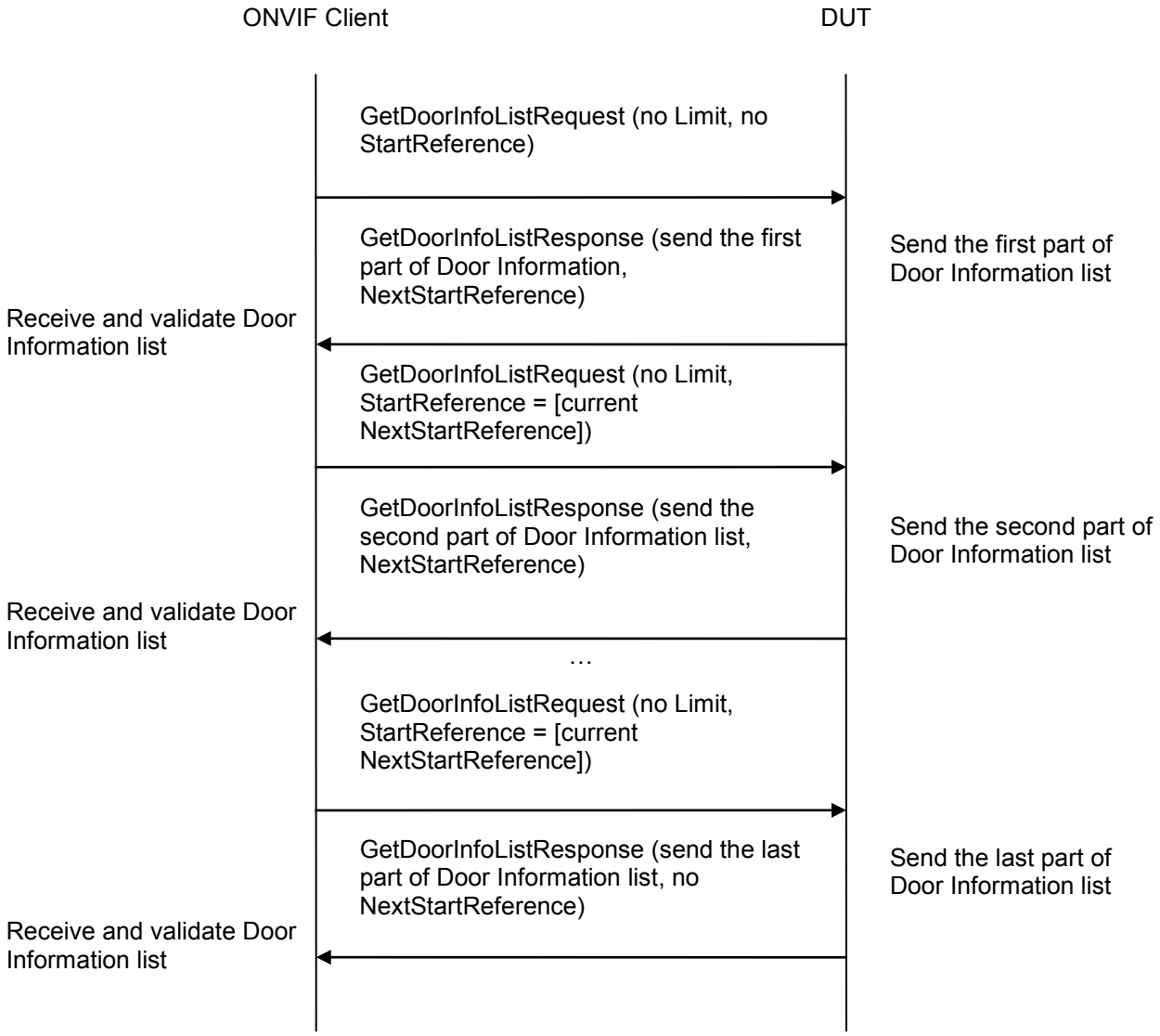
This section describes the meaning of the following definitions. These definitions are used in the test case description.

A.1 Get Complete Door Info List

The following algorithm will be used to get a complete list of Doors:

1. ONVIF Client will invoke GetDoorInfoListRequest message (**no Limit, no StartReference**) to retrieve the first part of Door Information list from the DUT.
2. Verify the GetDoorInfoListResponse message from the DUT.
3. If GetDoorInfoListResponse message contains NextStartReference, repeat steps 1-2 with **StartReference = [current NextStartReference]**. Otherwise, skip other steps and finalize getting complete door list.

The complete ordered list of doors with information will be made by the means of uniting all GetDoorInfoListResponse messages. Also, the total number of doors will be calculated.

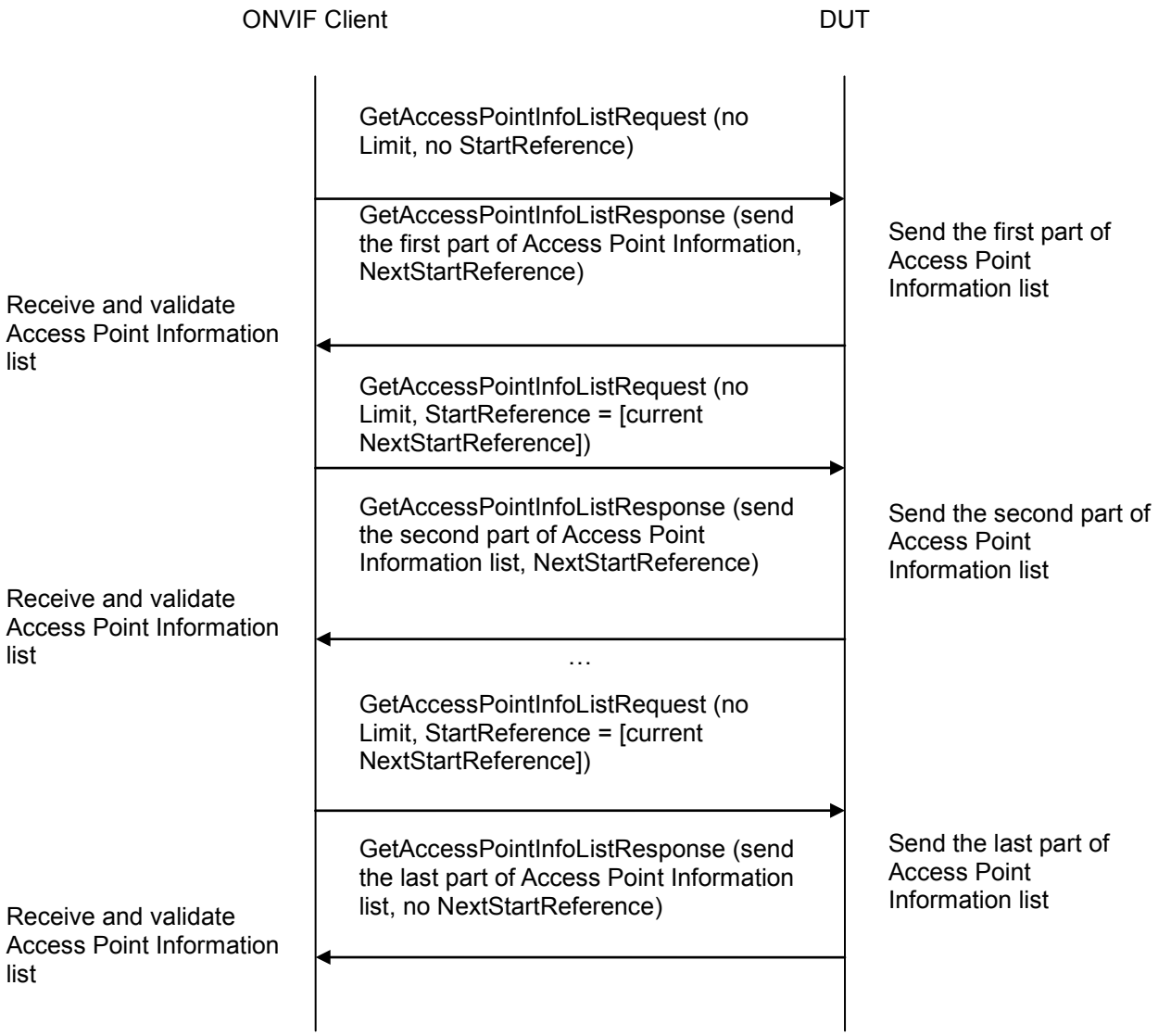


A.2 Get Complete Access Point Info List

The following algorithm will be used to get the 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 1-2 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 total number of access points will be calculated.



A.3 Catching of Initialized event

According to ONVIF-Core-Specification (9.4 Properties Item) device shall provide notifications informing the client of all objects with the requested property, which are alive at the time of the subscription. Therefore, after CreatePullPointSubscription request in DOORCONTROL-3-1-28 – DOORCONTROL-3-1-36 tests device shall generate Initialized events with current Door State for all existing doors.

ONVIF Client will follow the following logic to catch Initialized event for required Door token:

1. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
2. Verify that the DUT sends a PullMessagesResponse that contains NotificationMessages. If no NotificationMessage for selected Door is received and Operation Delay time does not expire, ONVIF Client repeats step 1.
3. Verify that the DUT sends Notification message for selected Door with current state (Notifications for other Doors will be ignored for this test case).
4. Verify received Notification message (correct value for UTC time, TopicExpression and wsnt:Message).
5. Verify that PropertyOperation = "Initialized".
6. Verify that TopicExpression is equal to tns1:Door/State/DoorMode.
7. Verify that notification contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to selected Door Token.
8. Verify that notification contains Data.SimpleItem item with Name = «State» and Value with type is equal to tdc:DoorMode.

FAIL –

The DUT did not send a valid Notification message that contains a property event tns1:Door/State/DoorMode for selected Door during Operation delay time (correct value for UTC time, TopicExpression and wsnt:Message, PropertyOperation = "Initialized", Topic = “tns1:Door/State/DoorMode”, contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to selected Door Token, contains Data.SimpleItem item with Name = «State» and Value with type equal to tdc:DoorMode).

Note: Notifications for other Doors than selected will be ignored.

Note: Test will be failed, if no required Notification messages are received at step 3 during a certain period of time (Operation Delay should be used in ONVIF Device Test Tool).

A.4 User Interaction Process

1. ONVIF Client sends UnsubscribeRequest message to end current subscription if subscription manager has been created during test.
2. Verify UnsubscribeResponse message.
3. User Interaction window appears.

4. Test Operator puts Door into the appropriate state. At that state the command (used in the test), which changes the door state, shall be accepted by the DUT.

5. Returns to the test and use DoorToken from user Interaction window as token of selected Door.

A.5 Door's transition to Locked state

ONVIF Client will try to move the selected for the test door into 'Locked' state according to the following logics:

1. ONVIF Client will invoke GetDoorStateRequest message (**Token = "Token1"**, where Token1 is selected door) to retrieve current state of the door.

2. Verify GetDoorStateResponse from the DUT.

3. If **DoorState.DoorMode** is equal to **Locked** then skip other steps and returns to the test.

4. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Door/State/DoorMode Topic as Filter and an InitialTerminationTime of timeout1.

5. Verify CreatePullPointSubscriptionResponse message from the DUT.

6. If **DoorState.DoorMode** is equal to **Unlocked** then ONVIF Client will invoke LockDoorRequest message (Token = "Token1") to switch the door into Locked state. Otherwise, go to step 12.

7. Verify LockDoorResponse message from the DUT.

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

9. Verify that the DUT sends a PullMessagesResponse. Repeat step 8 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).

10. Check that the last Notification message with PropertyOperation = "Changed" contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to **Token1** and contains Data.SimpleItem item with Name = «State» and Value equal to "**Locked**".

11. ONVIF Client deletes Subscription Manager either by calling unsubscribe or through a timeout and returns to the test.

12. If **DoorState.DoorMode** is equal to **Accessed** then ONVIF Client will invoke LockDoorRequest message (Token = "Token1") to switch the door into Locked state. Otherwise, go to step 18.

13. Verify LockDoorResponse message from the DUT.

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. Repeat step 14 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).

16. Check that the last Notification message with PropertyOperation = "Changed" contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to **Token1** and contains Data.SimpleItem item with Name = «State» and Value equal to "**Locked**".

17. ONVIF Client deletes Subscription Manager either by calling unsubscribe or through a timeout and returns to the test.

18. If **DoorState.DoorMode** is equal to **LockedDown** then ONVIF Client will invoke LockDownReleaseDoorRequest message (Token = “Token1”) to switch the door into Locked state. Otherwise, go to step 24.

19. Verify LockDownReleaseDoorResponse message from the DUT.

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. Repeat step 20 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).

22. Check that the last Notification message with PropertyOperation = "Changed" contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to **Token1** and contains Data.SimpleItem item with Name = «State» and Value equal to “**Locked**”.

23. ONVIF Client deletes Subscription Manager either by calling unsubscribe or through a timeout and returns to the test.

24. If **DoorState.DoorMode** is equal to **LockedOpen** then ONVIF Client will invoke LockOpenReleaseDoorRequest message (Token = “Token1”) to move the door into Unlocked state. Otherwise go to step 35.

25. Verify LockOpenReleaseDoorResponse message from the DUT.

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. Repeat step 26 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).

28. Check that the last Notification message with PropertyOperation = "Changed" contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to **Token1** and contains Data.SimpleItem item with Name = «State» and Value equal to “**Unlocked**”.

29. ONVIF Client will invoke LockDoorRequest message (Token = “Token1”) to switch the door into Locked state.

30. Verify LockDoorResponse message from the DUT.

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

32. Verify that the DUT sends a PullMessagesResponse. Repeat step 31 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).

33. Check that the last Notification message with PropertyOperation = "Changed" contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to **Token1** and contains Data.SimpleItem item with Name = «State» and Value equal to “**Locked**”.

34. ONVIF Client deletes Subscription Manager either by calling unsubscribe or through a timeout and returns to the test.

35. If **DoorState.DoorMode** is equal to **DoubleLocked** then ONVIF Client will invoke LockDoorRequest message (Token = “Token1”) to move the door into Locked state. Otherwise, go to step 41.

36. Verify LockDoorResponse message from the DUT.
37. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
38. Verify that the DUT sends a PullMessagesResponse. Repeat step 37 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
39. Check that the last Notification message with PropertyOperation = "Changed" contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to **Token1** and contains Data.SimpleItem item with Name = «State» and Value equal to "**Locked**".
40. ONVIF Client deletes Subscription Manager either by calling unsubscribe or through a timeout and returns to the test.
41. If **DoorState.DoorMode** is equal to **Blocked** then ONVIF Client will invoke LockDoorRequest message (Token = "Token1") to switch the door into Locked state. Otherwise, skip other steps and run user interaction process.
42. Verify LockDoorResponse message from the DUT.
43. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1.
44. Verify that the DUT sends a PullMessagesResponse. Repeat step 14 until timeout expires (Operation Delay option from ONVIF Device Test Tool will be used).
45. Check that the last Notification message with PropertyOperation = "Changed" contains Source.SimpleItem item with Name = "DoorToken" and Value is equal to **Token1** and contains Data.SimpleItem item with Name = «State» and Value equal to "**Locked**".
46. ONVIF Client deletes Subscription Manager either by calling unsubscribe or through a timeout and returns to the test.

FAIL –

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT sent the last Notification message with PropertyOperation = "Changed" with State value not equal to expected state at step 9 or 15 or 32 or 38.

The DUT sent an invalid Notification messages with PropertyOperation = "Changed" (correct value for UTC time, TopicExpression and wsnt:Message, Topic = "tns1:Door/State/DoorMode", contains Source.SimpleItem item with Name = "DoorToken" and Value equal to selected Door Token, contains Data.SimpleItem item with Name = «State» and Value with type equal to tdc:DoorMode).

Note: If the DUT sends SOAP fault to request to change door mode, then the ONVIF Client starts user interaction process (see Annex A.4).

Note: If DUT sends the last notification message with PropertyOperation = "Changed" with door state different from expected at step 21 or at step 27, then the ONVIF Client returns to the test and starts user interaction process (see Annex A.4).