Driving IP-based physical security through global standardization

ONVIF^{TM}

Receiver Test Specification

Version 13.12

December, 2013

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

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

Revision History

Ver.	Date	Description
12.12	20th/Dec, 2012	First issue of Receiver Test Specification
13.06	Jun, 2013	Update with spelling and typos fixes.
13.12	Dec, 2013	RECEIVER-2-1-14 CONFIGURE RECEIVER test was replased with RECEIVER-2-1-20 CONFIGURE RECEIVER RECEIVER-2-1-15 CONFIGURE RECEIVER – PERSISTENCE test was replased with RECEIVER-2-1-21 CONFIGURE RECEIVER – PERSISTENCE

Table of Contents

1 Introduction

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

The principal intended purposes are:

1. 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 will cover its subset.

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

1.1.1 Capabilities

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

1.1.2 General

General covers the test cases needed for the verification of receiver features as mentioned in [ONVIF Network Interface Specs]. General section defines different receiver test for getting of information about current receivers configuration and basic configuration of receivers.

2 Terms and Definitions

2.1 Definitions

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

Metadata	All streaming data except video and audio, including video analytics results, PTZ position data and other metadata (such as textual data from POS applications).

2.2 Abbreviations

This section describes abbreviations used in this document.

DUT	Device Under Test
HTTP	Hyper Text Transport Protocol
RTCP	RTP Control Protocol
RTSP	Real Time Streaming Protocol
RTP	Real-time Transport Protocol
SDP	Session Description Protocol
TCP	Transport Control Protocol
UTC	Coordinated Universal Time
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
WSDL	Web Services Description Language
WS-I BP 2.0	Web Services Interoperability Basic Profile version 2.0
XML	eXtensible Markup Language

3 Test Overview

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

3.1 Test Setup

3.1.1 Network Configuration for device under test

The generic test configuration for the execution of test cases defined in this document is as shown below (Figure 1)

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

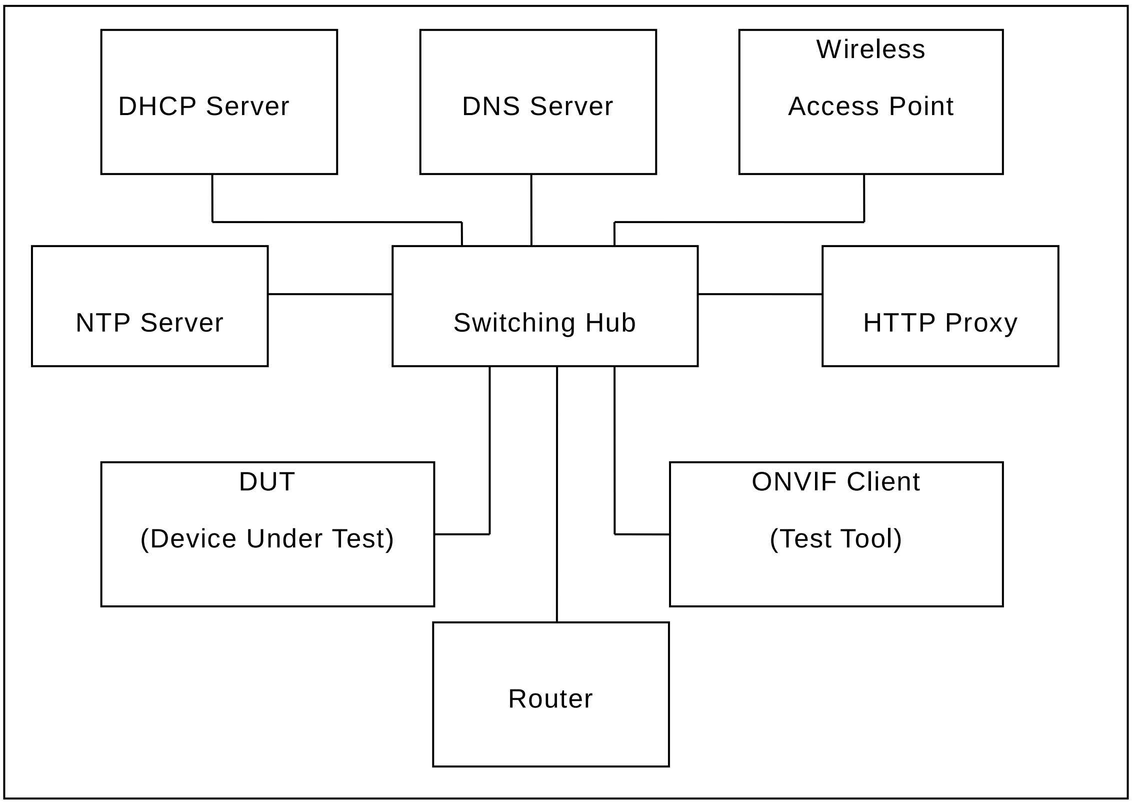


Figure 1: Test Configuration for DUT

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

ONVIF Client (Test Tool): Tests are executed by this system and it controls the 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 connectivity 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 Client 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 client Test tool time shall be synchronized with each other either manually or by common NTP server.

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

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

Please refer to Section 4.1 for Capabilities Test Cases.

3.3.2 General

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

Please refer to Section 4.2 for General Test Cases.

3.3.3 Authentication method selection as a testing framework

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

Procedure:

1. ONVIF Client invokes a specific command which is under testing without any user credentials (no WS-UsernameToken, no HTTP digest authentication header).

2. If DUT returns correct response, then ONVIF Client determines that DUT does not require any user authentication toward the command according to the configured security policy.

3. If DUT returns HTTP 401 Unauthorized error along with WWW-Authentication: Digest header, then ONVIF Client determines that DUT supports HTTP digest authentication. ONVIF Client shall provide with the proper level of user credential to continue the test procedure.

4. If the DUT returns SOAP fault (Sender/NotAuthorized) message, then ONVIF Client determines that WS-UsernameToken is supported by DUT. ONVIF Client shall provide with the proper level of user credential to continue the test procedure.

4 Receiver Test Cases

4.1 Capabilities

4.1.1 RECEIVER SERVICE CAPABILITIES

Test Label: Receiver Capabilities Verification.

Test Case ID: RECEIVER-1-1-1

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

Command under test: GetServiceCapabilities (for Receiver Service)

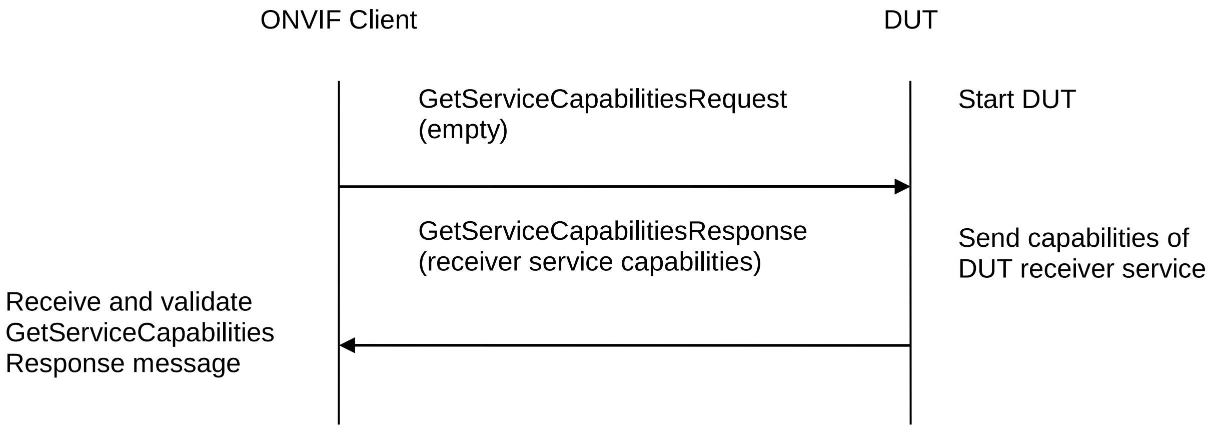
WSDL Reference: receiver.wsdl

Test Purpose: To verify DUT Receiver Service Capabilities.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:


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

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

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

4. Verify the GetServiceCapabilitiesResponse from the DUT.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetServiceCapabilitiesResponse.

The DUT sent Capabitites.MaximumRTSPURILength less than 128.

The DUT sent Capabitites.SupportedReceivers less than 1.

4.1.2 GET SERVICES AND GET RECEIVER SERVICE CAPABILITIES CONSISTENCY

Test Label: Get Services and Receiver Service Capabilities Consistency Verification.

Test Case ID: RECEIVER-1-1-2

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

Command under test: GetServices, GetServiceCapabilities (for Receiver Service)

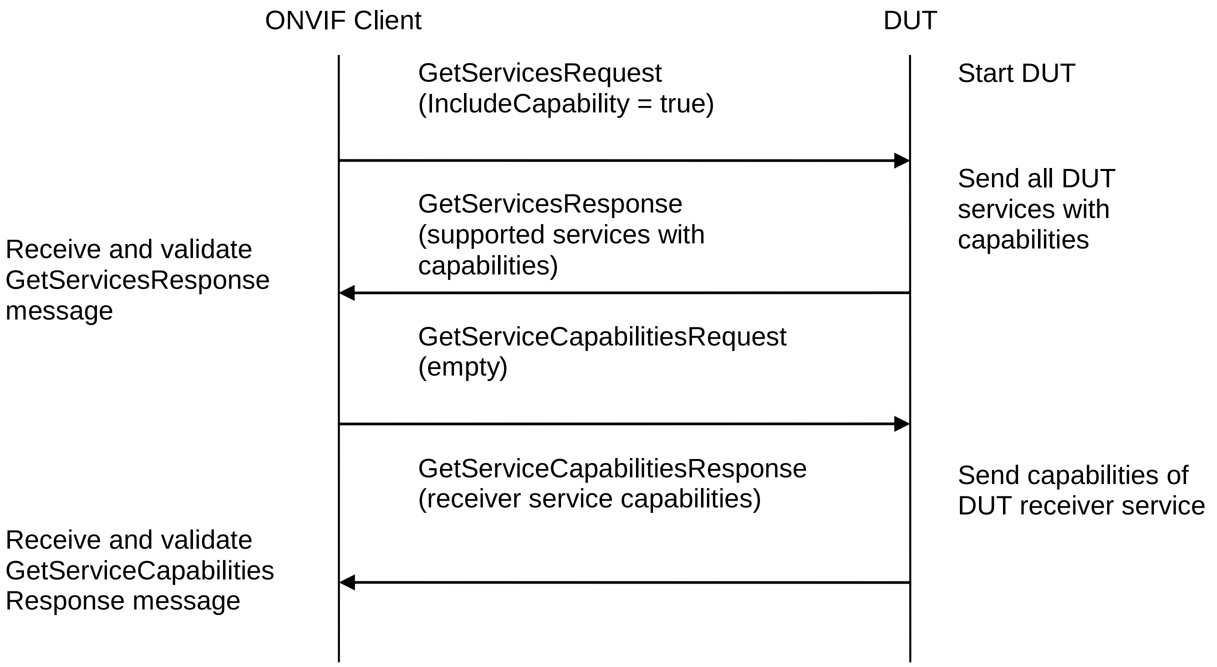
WSDL Reference: devicemgmt.wsdl, receiver.wsdl

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

Pre-Requisite: None.

Test Configuration: ONVIF Client and DUT

Test Sequence:


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

Test Procedure:

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

The DUT did not send a valid GetServiceCapabilitiesResponse message.

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

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

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

- RTP_Multicast attribute is skipped only for GetServicesResponse message or only for GetServiceCapabilitiesResponse message.
- RTP_Multicast attribute values are different.
- RTP_TCP attribute is skipped only for GetServicesResponse message or only for GetServiceCapabilitiesResponse message.
- RTP_TCP attribute values are different.
- RTP_RTSP_TCP attribute is skipped only for GetServicesResponse message or only for GetServiceCapabilitiesResponse message.
- RTP_RTSP_TCP attribute values are different.
- SupportedReceivers attribute values are different.
- MaximumRTSPURILength attribute is skipped only for GetServicesResponse message or only for GetServiceCapabilitiesResponse message.
- MaximumRTSPURILength attribute values are different.

4.2 General

4.2.1 GET RECEIVERS

Test Label: Get Receivers Verification.

Test Case ID: RECEIVER-2-1-1

ONVIF Core Specification Coverage: GetReceivers (ONVIF Receiver Service Specification)

Command under test: GetReceivers

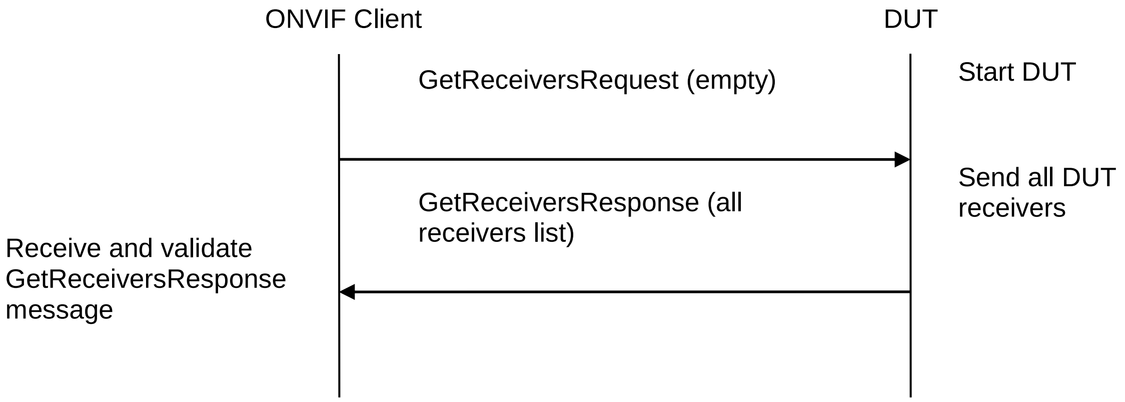
WSDL Reference: receiver.wsdl

Test Purpose: To verify Get Receivers.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
GetReceiversRequest (empty)
Start DUT
Send all DUT receivers
GetReceiversResponse (all receivers list)
Receive and validate GetReceiversResponse message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.

4. Verify the GetReceiversResponse message from the DUT.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetReceiversResponse message.

The DUT sent at least two Receivers with the same Receivers.Token in the GetReceiversResponse message.

4.2.2 GET RECEIVER

Test Label: Get Receiver Verification.

Test Case ID: RECEIVER-2-1-2

ONVIF Core Specification Coverage: GetReceiver (ONVIF Receiver Service Specification)

Command under test: GetReceiver

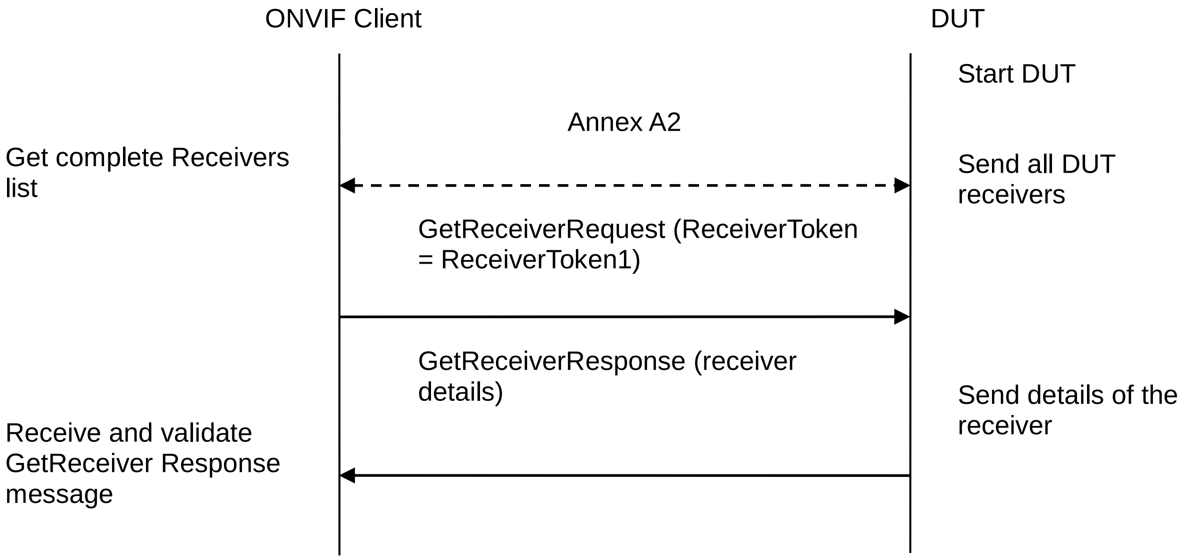
WSDL Reference: receiver.wsdl

Test Purpose: To verify Get Receiver.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A2
Get complete Receivers list
Send all DUT receivers
GetReceiverRequest (ReceiverToken = ReceiverToken1)
GetReceiverResponse (receiver details)
Send details of the receiver
Receive and validate GetReceiver Response message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).

4. ONVIF Client will invoke GetReceiverRequest message (**ReceiverToken = “ReceiverToken1”**, where ReceiverToken1 is the first **Receivers.Token** from the GetReceiversResponse message at step at step 3) to retrieve receiver configuration.

5. Verify the GetReceiverResponse message from the DUT.

6. Repeat steps 5-6 for all other receivers from the GetReceiversResponse message at step at step 3.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send a valid GetReceiverResponse message.

The DUT returned different parameter values for the same receiver in GetReceiversResponse message and in GetReceiverResponse message (see Annex A.1).

Note: Two Receivers with the same token shall have the same properties.

4.2.3 GET RECEIVER WITH INVALID TOKEN

Test Label: Get Receiver Verification with Invalid Token.

Test Case ID: RECEIVER-2-1-3

ONVIF Core Specification Coverage: GetReceiver (ONVIF Receiver Service Specification)

Command under test: GetReceiver

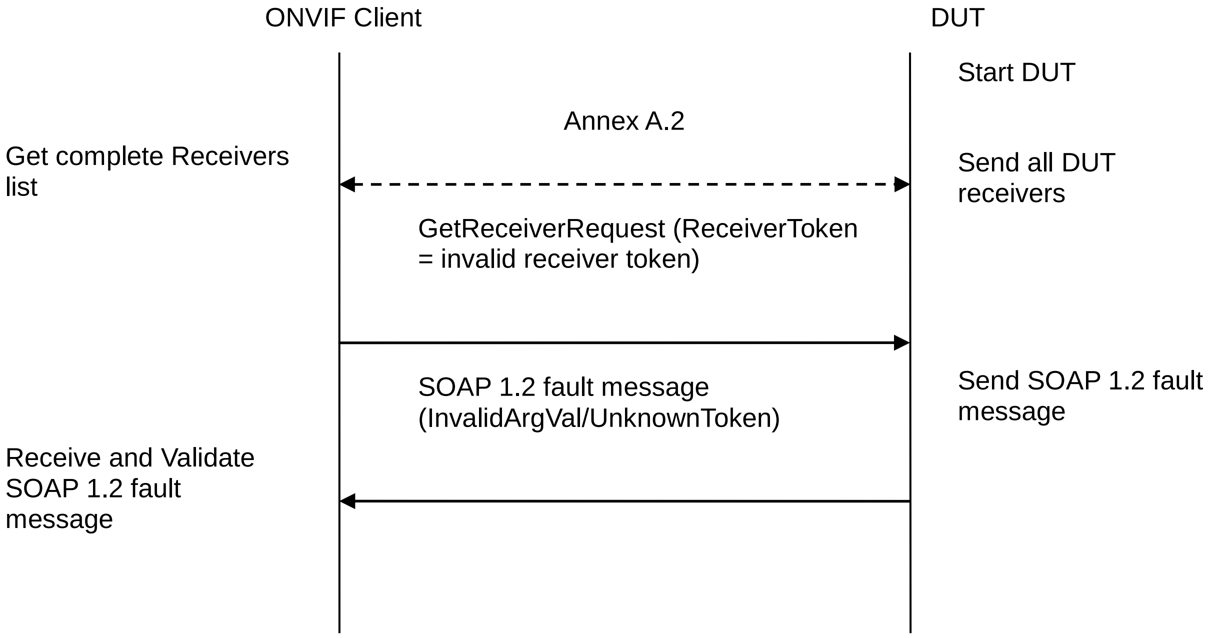
WSDL Reference: receiver.wsdl

Test Purpose: To verify Get Receiver with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete Receivers list
Send all DUT receivers
GetReceiverRequest (ReceiverToken = invalid receiver token)
SOAP 1.2 fault message (InvalidArgVal/UnknownToken)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).
4. ONVIF Client will invoke GetReceiverRequest message **(invalid ReceiverToken**).
5. The DUT will generate SOAP 1.2 fault message (**InvalidArgVal/UnknownToken**).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send SOAP 1.2 fault message.

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

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

4.2.4 GET RECEIVER STATE

Test Label: Get Receiver State Verification.

Test Case ID: RECEIVER-2-1-4

ONVIF Core Specification Coverage: GetReceiverState (ONVIF Receiver Service Specification)

Command under test: GetReceiverState

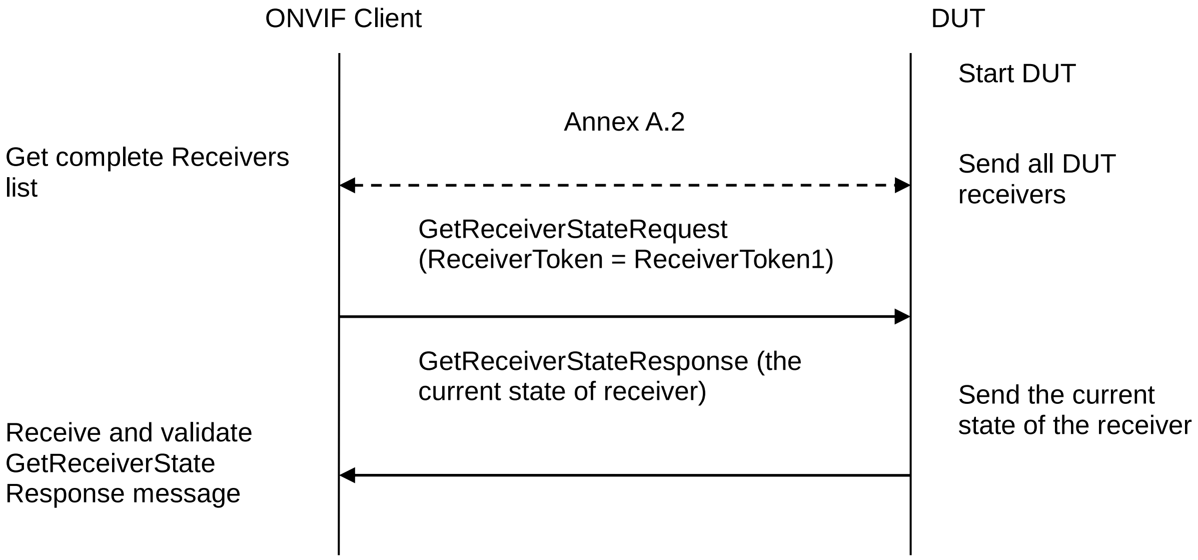
WSDL Reference: receiver.wsdl

Test Purpose: To verify Get Receiver State.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete Receivers list
Send all DUT receivers
GetReceiverStateRequest (ReceiverToken = ReceiverToken1)
GetReceiverStateResponse (the current state of receiver)
Send the current state of the receiver
Receive and validate GetReceiverState Response message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).

4. ONVIF Client will invoke GetReceiverStateRequest message (**ReceiverToken = "Token1"**, where Token1 is the first **Receivers.Token** from the GetReceiversResponse message at step at step 3) to retrieve receiver state.

5. Verify the GetReceiverStateResponse message from the DUT.

6. Repeat steps 5-6 for all other receivers from the GetReceiversResponse message at step at step 3.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send a valid GetReceiverStateResponse message.

4.2.5 GET RECEIVER STATE WITH INVALID TOKEN

Test Label: Get Receiver State Verification with Invalid Token.

Test Case ID: RECEIVER-2-1-5

ONVIF Core Specification Coverage: GetReceiverState (ONVIF Receiver Service Specification)

Command under test: GetReceiverState

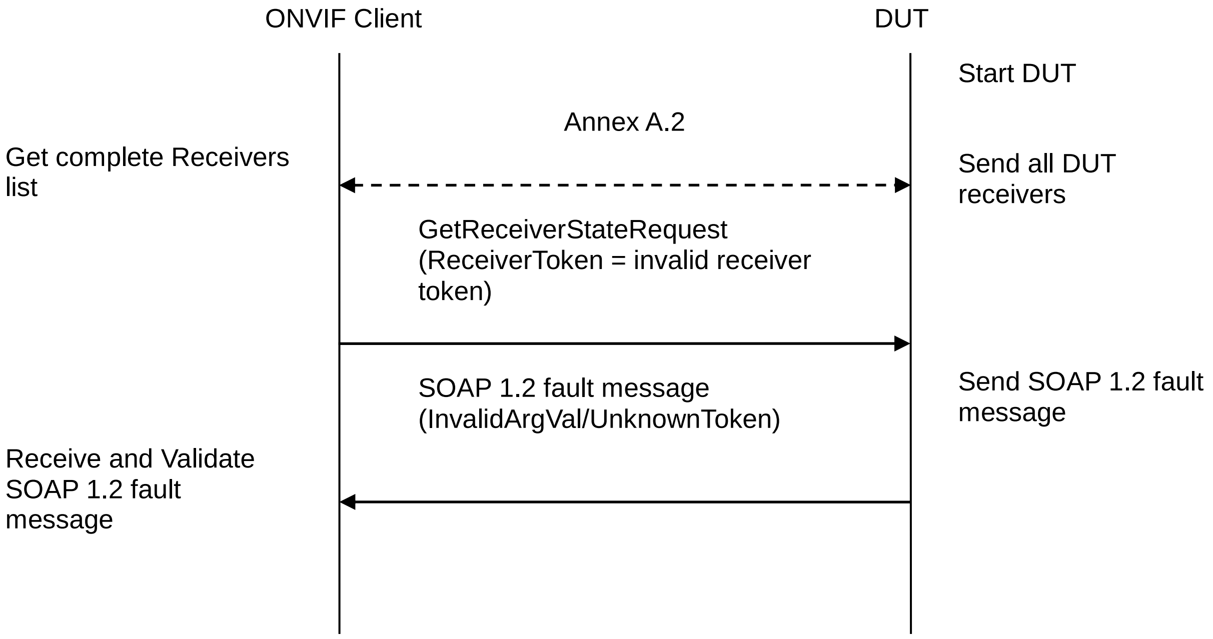
WSDL Reference: receiver.wsdl

Test Purpose: To verify Get Receiver State with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:



ONVIF Client
DUT
Start DUT
Annex A.2
Get complete Receivers list
Send all DUT receivers
GetReceiverStateRequest (ReceiverToken = invalid receiver token)
SOAP 1.2 fault message (InvalidArgVal/UnknownToken)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

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

3. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).

4. ONVIF Client will invoke GetReceiverStateRequest message **(invalid ReceiverToken**).

5. The DUT will generate SOAP 1.2 fault message (**InvalidArgVal/UnknownToken**).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send SOAP 1.2 fault message.

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

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

4.2.6 CREATE RECEIVER

Test Label: Create Receiver Verification.

Test Case ID: RECEIVER-2-1-6

ONVIF Core Specification Coverage: GetReceiver (ONVIF Receiver Service Specification), GetReceivers (ONVIF Receiver Service Specification), CreateReceiver (ONVIF Receiver Service Specification)

Command under test: GetReceiver, CreateReceiver, GetReceivers

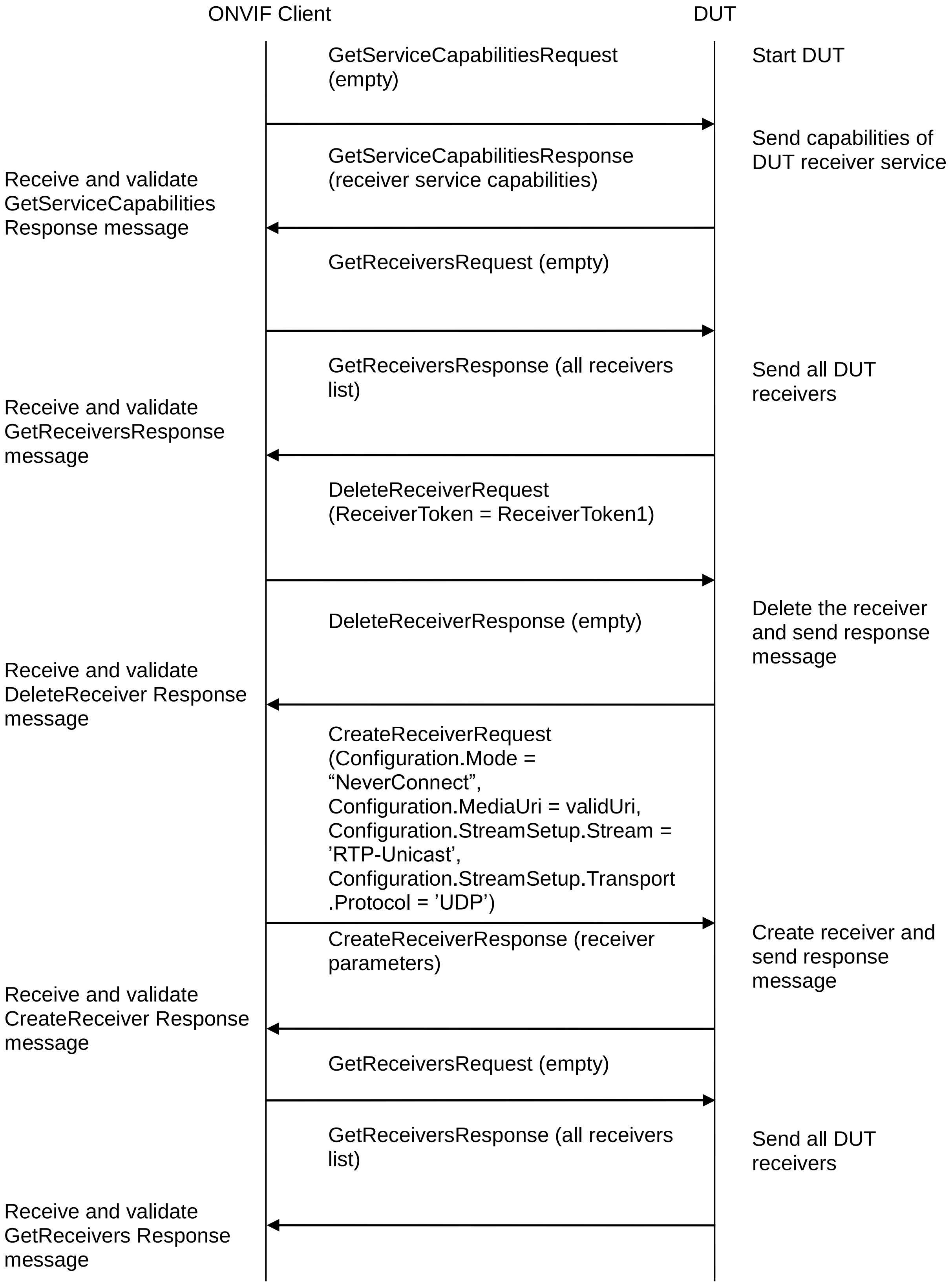
WSDL Reference: receiver.wsdl

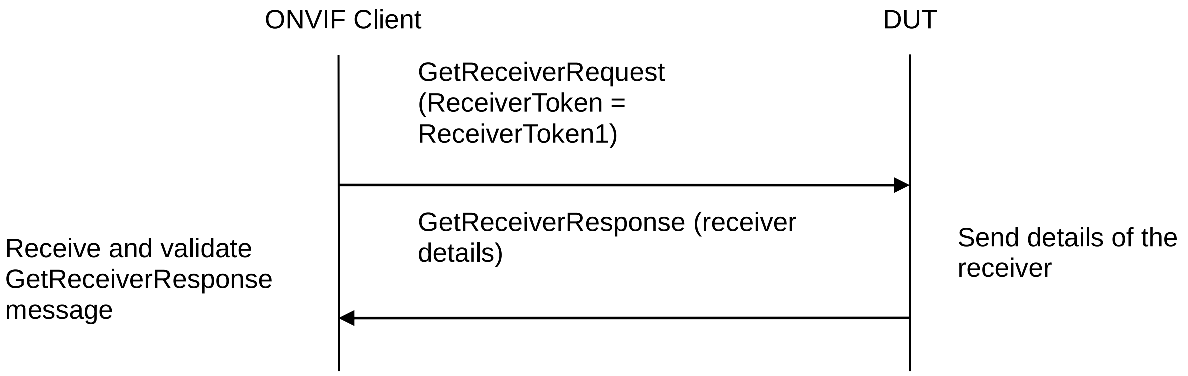
Test Purpose: To verify Create Receiver.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
GetServiceCapabilitiesRequest (empty)
Start DUT
GetServiceCapabilitiesResponse (receiver service capabilities)
Send capabilities of DUT receiver service
Receive and validate GetServiceCapabilities Response message
GetReceiversRequest (empty)
GetReceiversResponse (all receivers list)
Send all DUT receivers
Receive and validate GetReceiversResponse message
DeleteReceiverRequest (ReceiverToken = ReceiverToken1)
DeleteReceiverResponse (empty)
Delete the receiver and send response message
Receive and validate DeleteReceiver Response message
CreateReceiverRequest (Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport .Protocol = ’UDP’)
CreateReceiverResponse (receiver parameters)
Create receiver and send response message
Receive and validate CreateReceiver Response message
GetReceiversRequest (empty)
GetReceiversResponse (all receivers list)
Send all DUT receivers
Receive and validate GetReceivers Response message


ONVIF Client
DUT
GetReceiverRequest (ReceiverToken = ReceiverToken1)
GetReceiverResponse (receiver details)
Receive and validate GetReceiverResponse message
Send details of the receiver

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

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

4. Verify the GetServiceCapabilitiesResponse message (Capabilities.SupportedReceivers) from the DUT.

5. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.

6. Verify the GetReceiversResponse message from the DUT.

7. If number of Receivers from GetReceiversResponse message is less than specified in Capabilities.SupportedReceivers then skip steps 8-9 and go to step 10

8. ONVIF Client will invoke DeleteReceiverRequest message (ReceiverToken = **ReceiverToken1**) to delete Receiver.

9. Verify the DeleteReceiverResponse message from the DUT.

10. ONVIF Client will invoke CreateReceiverRequest message (Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport.Protocol = ’UDP’) to create new Receiver.

11. Verify the CreateReceiverResponse message (Token = token2, Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport.Protocol = ’UDP’) from the DUT. Check that CreateReceiverResponse message contains the same parameters values as was sent in CreateReceiverRequest message.

12. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.

13. Verify the GetReceiversResponse message (receivers list with new receiver) from the DUT. Check that Receiver was created with specified parameters.

14. ONVIF Client will invoke GetReceiverRequest message (Token = token2) to retrieve receiver configuration.
15. Verify the GetReceiverResponse message (Token = token2, Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport.Protocol = ’UDP’) from the DUT. Check that Receiver was created with the specified parameters.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetServiceCapabilitiesResponse message.

The DUT did not send a valid DeleteReceiverResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send a valid GetReceiverResponse message.

The DUT did not send a valid GetReceiversResponse message.

The DUT sent GetReceiverResponse message without created Receiver for step 13.

The DUT returned Receiver parameters in GetReceiversResponse message that differ from the specified during Receiver creation.

The DUT returned Receiver parameters in GetReceiverResponse message that differ from the specified during Receiver creation.

The DUT returned Receiver parameters in CreateReceiverResponse message that differ from the specified during Receiver creation.

Note: If the fault (Action/CannotDeleteReceiver) was received during receiver deletion for step 8, try to delete some other Receiver. If there are no Receivers that could not be deleted, skip steps 10-15 and go to the next test.

4.2.7 CREATE RECEIVER – PERSISTENCE

Test Label: Create Receiver Persistence Verification.

Test Case ID: RECEIVER-2-1-7

ONVIF Core Specification Coverage: CreateReceiver (ONVIF Receiver Service Specification), Persistence (ONVIF Receiver Service Specification)

Command under test: CreateReceiver

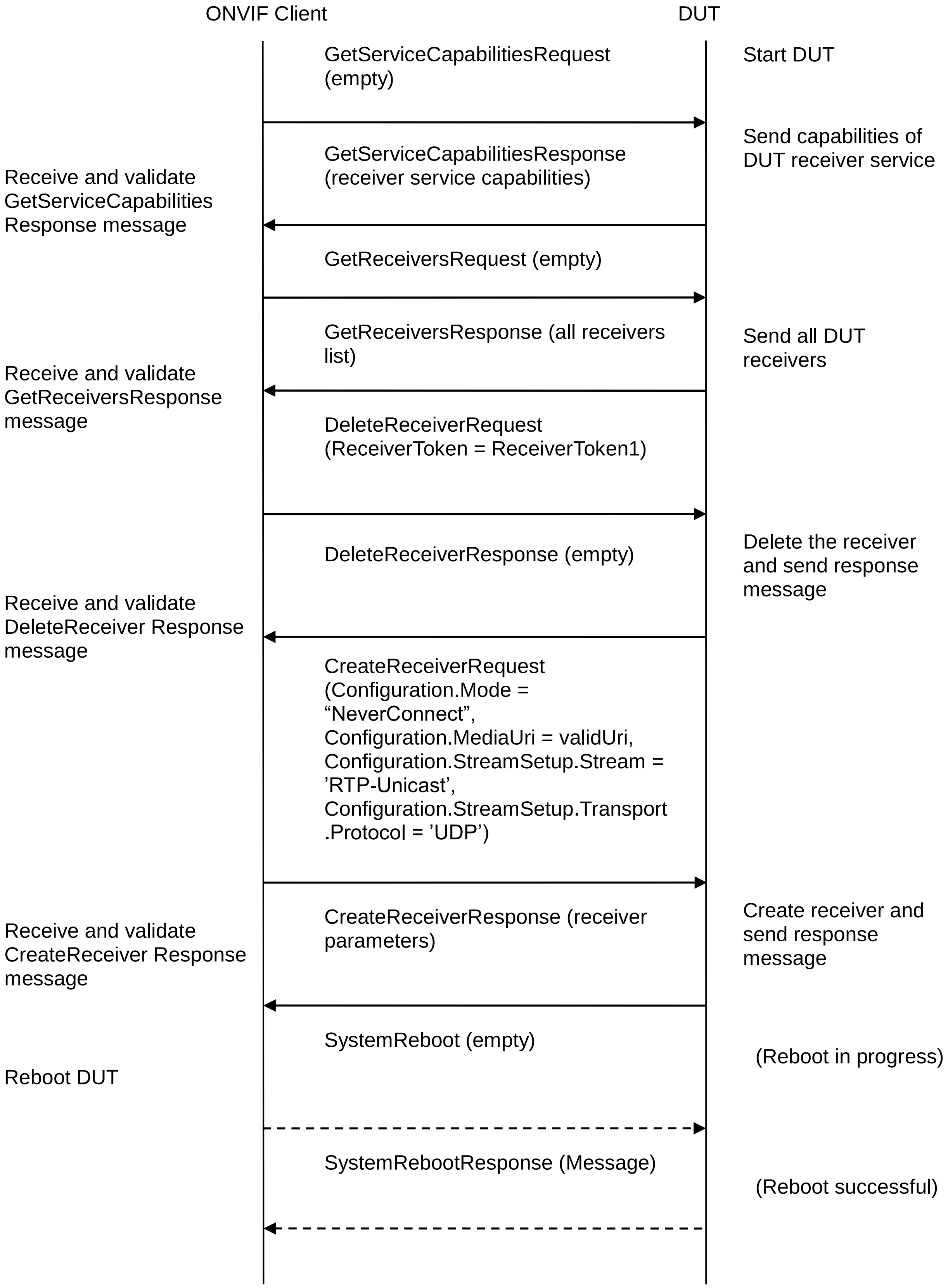
WSDL Reference: receiver.wsdl

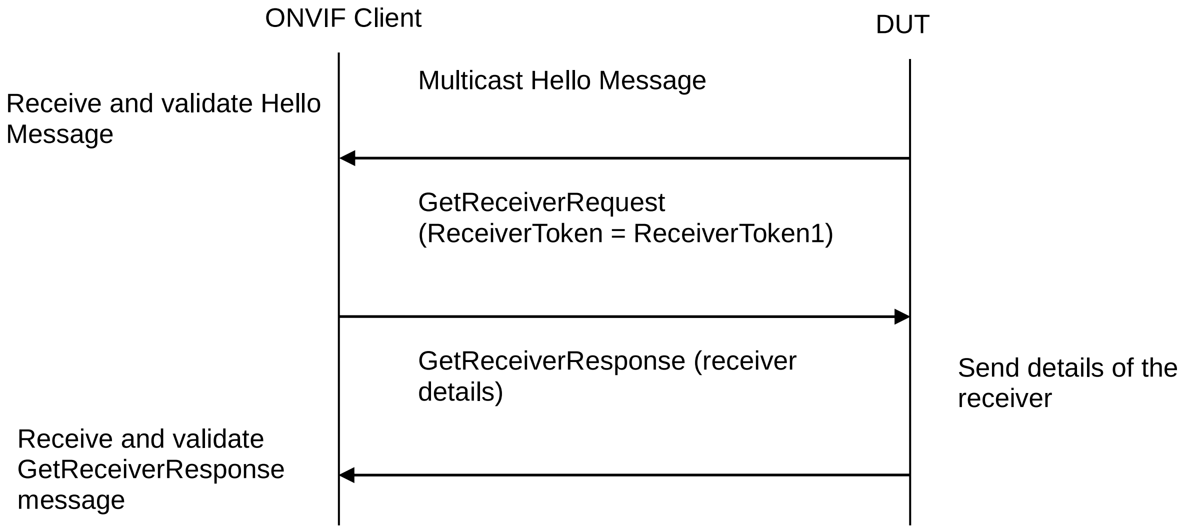
Test Purpose: To verify Create Receiver Persistence.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
GetServiceCapabilitiesRequest (empty)
Start DUT
GetServiceCapabilitiesResponse (receiver service capabilities)
Send capabilities of DUT receiver service
Receive and validate GetServiceCapabilities Response message
GetReceiversRequest (empty)
GetReceiversResponse (all receivers list)
Send all DUT receivers
Receive and validate GetReceiversResponse message
DeleteReceiverRequest (ReceiverToken = ReceiverToken1)
DeleteReceiverResponse (empty)
Delete the receiver and send response message
Receive and validate DeleteReceiver Response message
CreateReceiverRequest (Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport .Protocol = ’UDP’)
CreateReceiverResponse (receiver parameters)
Create receiver and send response message
Receive and validate CreateReceiver Response message
SystemReboot (empty)
Reboot DUT
(Reboot in progress)
SystemRebootResponse (Message)
(Reboot successful)


ONVIF Client
DUT
Multicast Hello Message
Receive and validate Hello Message
GetReceiverRequest (ReceiverToken = ReceiverToken1)
GetReceiverResponse (receiver details)
Send details of the receiver
Receive and validate GetReceiverResponse message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

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

4. Verify the GetServiceCapabilitiesResponse message (Capabilities.SupportedReceivers) from the DUT.

5. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.

6. Verify the GetReceiversResponse message from the DUT.

7. If number of Receivers from GetReceiversResponse message is less than specified in Capabilities.SupportedReceivers, then skip steps 8-9 and go to step 10

8. ONVIF Client will invoke DeleteReceiverRequest message (ReceiverToken = **ReceiverToken1**) to delete Receiver.

9. Verify the DeleteReceiverResponse message from the DUT.

10. ONVIF Client will invoke CreateReceiverRequest message (Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport.Protocol = ’UDP’) to create new Receiver.

11. Verify the CreateReceiverResponse message (Token = token2, Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport.Protocol = ’UDP’) from the DUT. Check that CreateReceiverResponse message contains the same parameters values as was sent in CreateReceiverRequest message.

12. ONVIF Client will invoke SystemReboot message to reset the DUT.

13. Verify that DUT sends SystemRebootResponse message (example message string = “Rebooting in x seconds”).
14. DUT will send Multicast HELLO message after it is successfully rebooted.
15. ONVIF Client will verify the HELLO message sent by DUT.
16. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.
17. Verify the GetReceiversResponse message (receivers list with new receiver) from the DUT. Check that Receiver was created with specified parameters.
18. ONVIF Client will invoke GetReceiverRequest message (Token = token2) to retrieve receiver configuration.
19. Verify the GetReceiverResponse message (Token = token2, Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport.Protocol = ’UDP’) from the DUT. Check that Receiver was created with the specified parameters.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetServiceCapabilitiesResponse message.

The DUT did not send a valid DeleteReceiverResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send a valid GetReceiverResponse message.

The DUT did not send a valid GetReceiversResponse message.

The DUT sent GetReceiversResponse message without created Receiver for step 13.

The DUT returned Receiver parameters in GetReceiversResponse message that differ from the specified during Receiver creation.

The DUT returned Receiver parameters in GetReceiverResponse message that differ from the specified during Receiver creation.

The DUT returned Receiver parameters in CreateReceiverResponse message that differ from the specified during Receiver creation.

The DUT did not send SystemRebootResponse message.

The DUT did not send HELLO message.

Note: In case a fault (Action/CannotDeleteReceiver) was received during receiver deletion for step 8, try to delete other Receiver. If there are no Receivers that could not be deleted, skip steps 10-15 and go to the next test.

4.2.8 CREATE RECEIVER – RECEIVERS MAX NUMBER

Test Label: Maximum Number of Receivers Creation Verification.

Test Case ID: RECEIVER-2-1-8

ONVIF Core Specification Coverage: GetServiceCapabilities (ONVIF Receiver Service Specification), CreateReceiver (ONVIF Receiver Service Specification)

Command under test: CreateReceiver

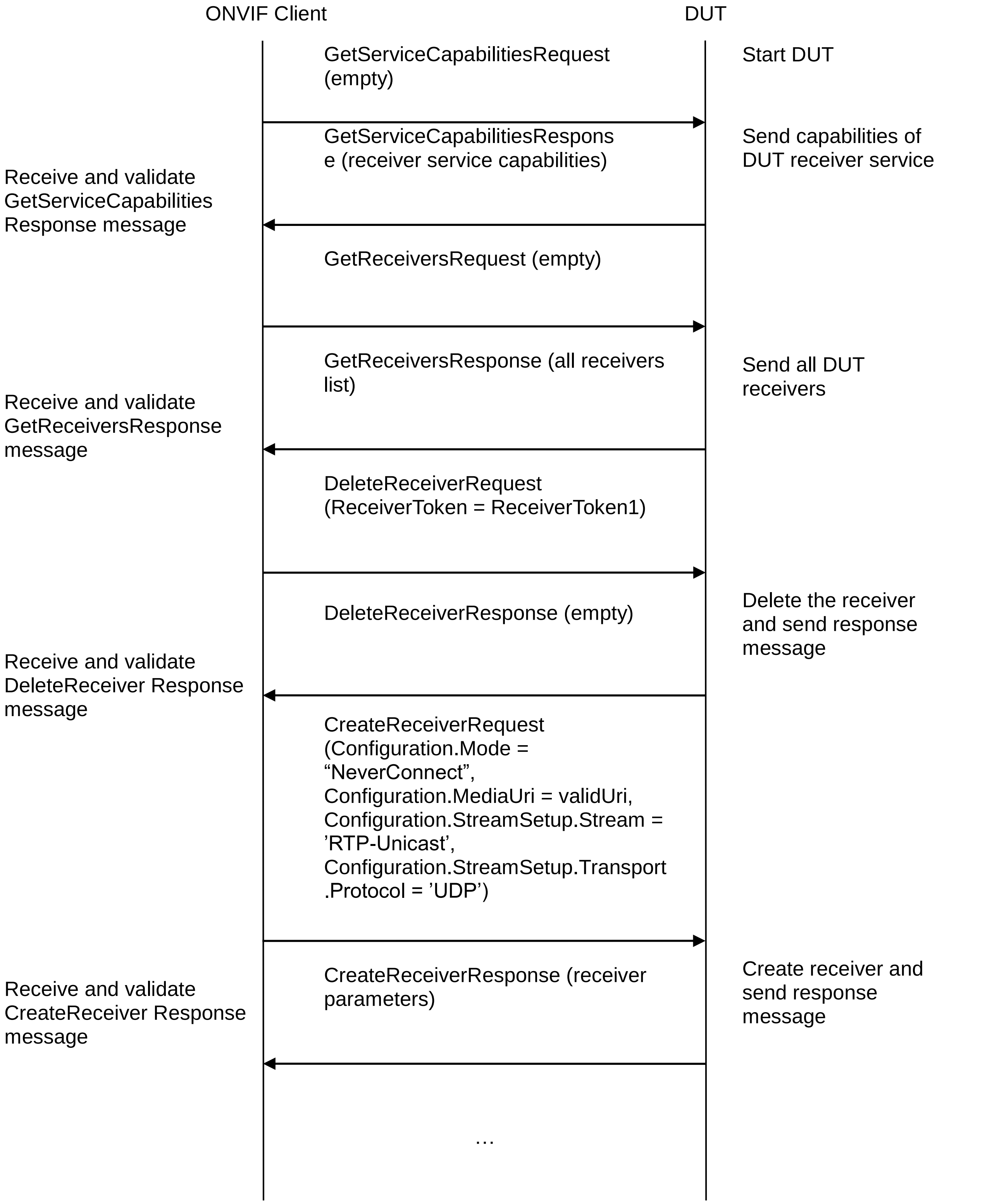
WSDL Reference: receiver.wsdl

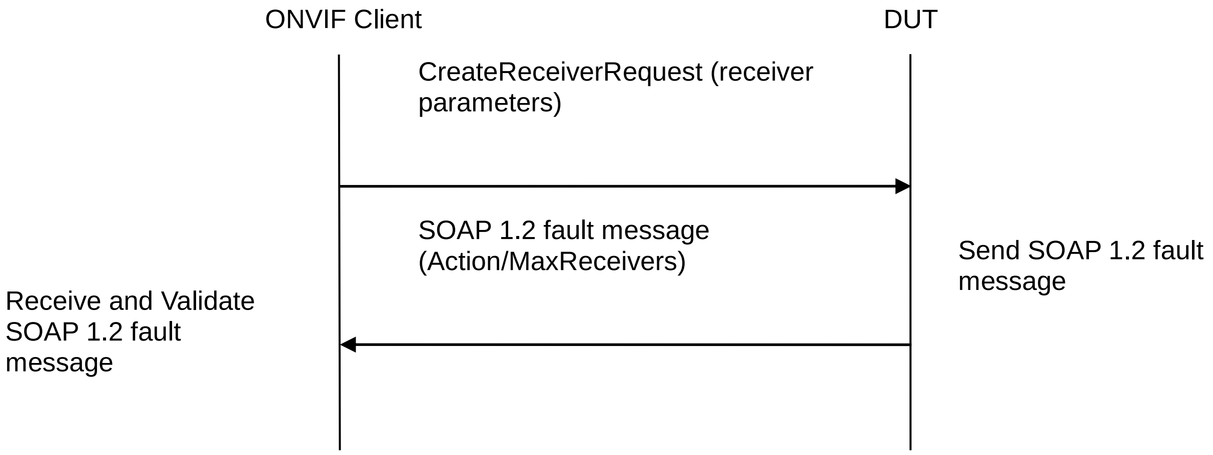
Test Purpose: To verify that the DUT supports specified maximum number of Receivers.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
GetServiceCapabilitiesRequest (empty)
Start DUT
GetServiceCapabilitiesResponse (receiver service capabilities)
Send capabilities of DUT receiver service
Receive and validate GetServiceCapabilities Response message
GetReceiversRequest (empty)
GetReceiversResponse (all receivers list)
Send all DUT receivers
Receive and validate GetReceiversResponse message
DeleteReceiverRequest (ReceiverToken = ReceiverToken1)
DeleteReceiverResponse (empty)
Delete the receiver and send response message
Receive and validate DeleteReceiver Response message
CreateReceiverRequest (Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport .Protocol = ’UDP’)
CreateReceiverResponse (receiver parameters)
Create receiver and send response message
Receive and validate CreateReceiver Response message
…


ONVIF Client
DUT
CreateReceiverRequest (receiver parameters)
SOAP 1.2 fault message (Action/MaxReceivers)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve receiver service capabilities of the DUT.
4. Verify the GetServiceCapabilitiesResponse message (Capabilities.SupportedReceivers) from the DUT.
5. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.
6. Verify the GetReceiversResponse message from the DUT.
7. If the number of Receivers from GetReceiversResponse message is equal to number that specified in Capabilities.SupportedReceivers then skip steps 8-9 and go to step 11
8. ONVIF Client will invoke CreateReceiverRequest message (Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport.Protocol = ’UDP’) to create new Receiver.
9. Verify the CreateReceiverResponse message (Token = token2, Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport.Protocol = ’UDP’) from the DUT. Check that CreateReceiverResponse message contains the same parameters values as was sent in CreateReceiverRequest message.
10. Repeat steps 7-9.
11. ONVIF Client will invoke CreateReceiverRequest message.
12. The DUT will generate SOAP 1.2 fault message (**Action/MaxReceivers**).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetServiceCapabilitiesResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send a valid GetReceiverResponse message.

The DUT did not send SOAP 1.2 fault message.

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

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

4.2.9 DELETE RECEIVER

Test Label: Delete Receiver Verification.

Test Case ID: RECEIVER-2-1-9

ONVIF Core Specification Coverage: GetReceiver (ONVIF Receiver Service Specification), GetReceivers (ONVIF Receiver Service Specification), DeleteReceiver (ONVIF Receiver Service Specification)

Command under test: GetReceiver, DeleteReceiver, GetReceivers

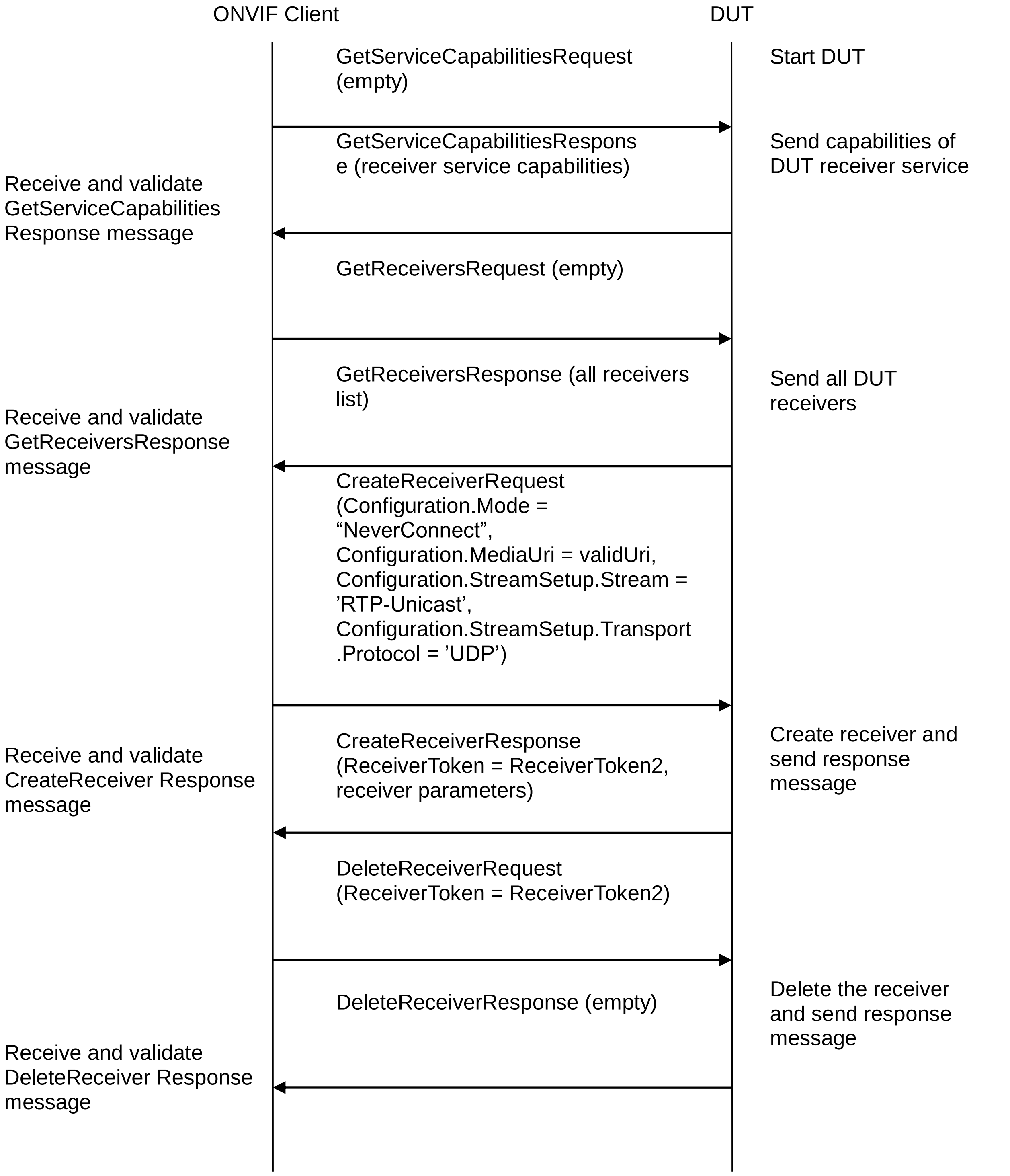
WSDL Reference: receiver.wsdl

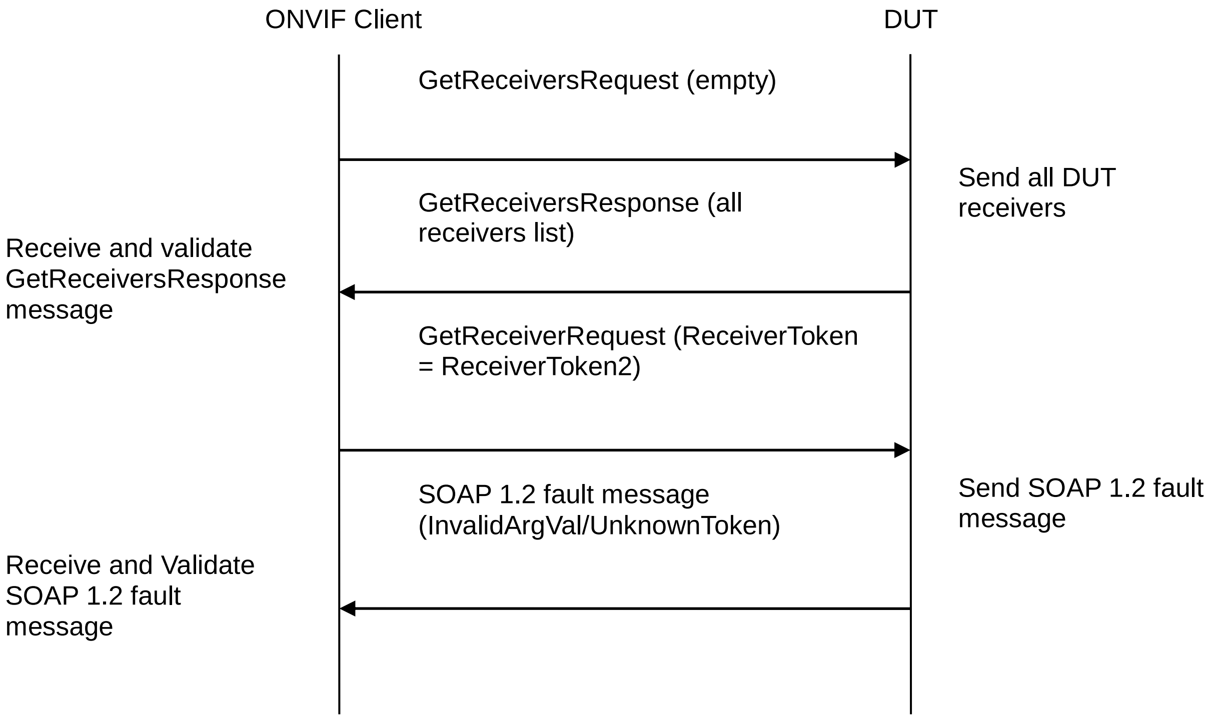
Test Purpose: To verify Delete Receiver.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
GetServiceCapabilitiesRequest (empty)
Start DUT
GetServiceCapabilitiesResponse (receiver service capabilities)
Send capabilities of DUT receiver service
Receive and validate GetServiceCapabilities Response message
GetReceiversRequest (empty)
GetReceiversResponse (all receivers list)
Send all DUT receivers
Receive and validate GetReceiversResponse message
CreateReceiverRequest (Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport .Protocol = ’UDP’)
CreateReceiverResponse (ReceiverToken = ReceiverToken2, receiver parameters)
Create receiver and send response message
Receive and validate CreateReceiver Response message
DeleteReceiverRequest (ReceiverToken = ReceiverToken2)
DeleteReceiverResponse (empty)
Delete the receiver and send response message
Receive and validate DeleteReceiver Response message


ONVIF Client
DUT
GetReceiversRequest (empty)
Send all DUT receivers
GetReceiversResponse (all receivers list)
Receive and validate GetReceiversResponse message
GetReceiverRequest (ReceiverToken = ReceiverToken2)
SOAP 1.2 fault message (InvalidArgVal/UnknownToken)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve receiver service capabilities of the DUT.
4. Verify the GetServiceCapabilitiesResponse message (Capabilities.SupportedReceivers) from the DUT.
5. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.
6. Verify the GetReceiversResponse message from the DUT.
7. If the number of Receivers from GetReceiversResponse message is equal to what is specified in Capabilities.SupportedReceivers then skip steps 8-9 and go to step 10.
8. ONVIF Client will invoke CreateReceiverRequest message (Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport.Protocol = ’UDP’) to create new Receiver.
9. Verify the CreateReceiverResponse message (Token = **ReceiverToken2**, Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport.Protocol = ’UDP’) from the DUT. Check that CreateReceiverResponse message contains the same parameters values as was sent in CreateReceiverRequest message.

10. ONVIF Client will invoke DeleteReceiverRequest message (ReceiverToken = **ReceiverToken2**) to delete Receiver.

11. Verify the DeleteReceiverResponse message from the DUT.

12. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.

13. Verify the GetReceiversResponse message from the DUT. Check that Receiver (Token = Token2) was deleted and does not present in response message.

14. ONVIF Client will invoke GetReceiverRequest message (Token = ReceiverToken2).

15. The DUT will generate SOAP 1.2 fault message (**InalidArgVal/UnknownToken**).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetServiceCapabilitiesResponse message.

The DUT did not send a valid DeleteReceiverResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send a valid GetReceiverResponse message.

The DUT did not send a valid GetReceiversResponse message.

The DUT sent GetReceiverResponse message with created Receiver for step 14.

The DUT did not send SOAP 1.2 fault message for step 15.

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

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

4.2.10 DELETE RECEIVER WITH INVALID TOKEN

Test Label: Delete Receiver Verification with Invalid Token.

Test Case ID: RECEIVER-2-1-10

ONVIF Core Specification Coverage: DeleteReceiver (ONVIF Receiver Service Specification)

Command under test: DeleteReceiver

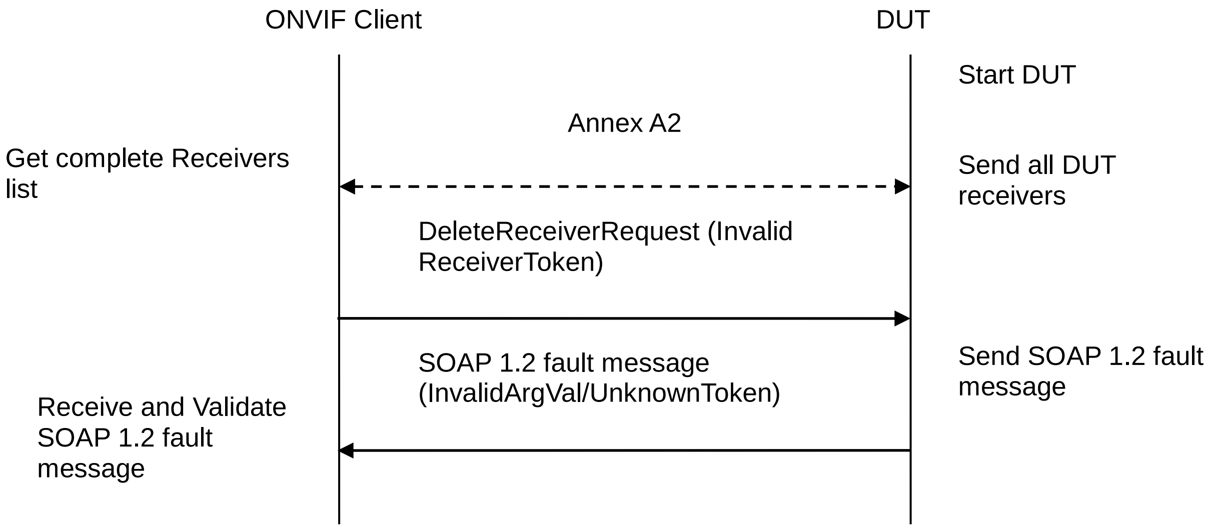
WSDL Reference: receiver.wsdl

Test Purpose: To verify Delete Receiver with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A2
Get complete Receivers list
Send all DUT receivers
DeleteReceiverRequest (Invalid ReceiverToken)
SOAP 1.2 fault message (InvalidArgVal/UnknownToken)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

1. Start an ONVIF Client.
2. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).
3. ONVIF Client will invoke DeleteReceiverRequest message **(invalid ReceiverToken**).
4. The DUT will generate SOAP 1.2 fault message (**InvalidArgVal/UnknownToken**).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send SOAP 1.2 fault message.

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

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

4.2.11 SET RECEIVER MODE

Test Label: Set Receiver Mode Verification.

Test Case ID: RECEIVER-2-1-11

ONVIF Core Specification Coverage: GetReceivers (ONVIF Receiver Service Specification), GetReceiver (ONVIF Receiver Service Specification), SetReceiverMode (ONVIF Receiver Service Specification)

Command under test: GetReceiver, SetReceiverMode

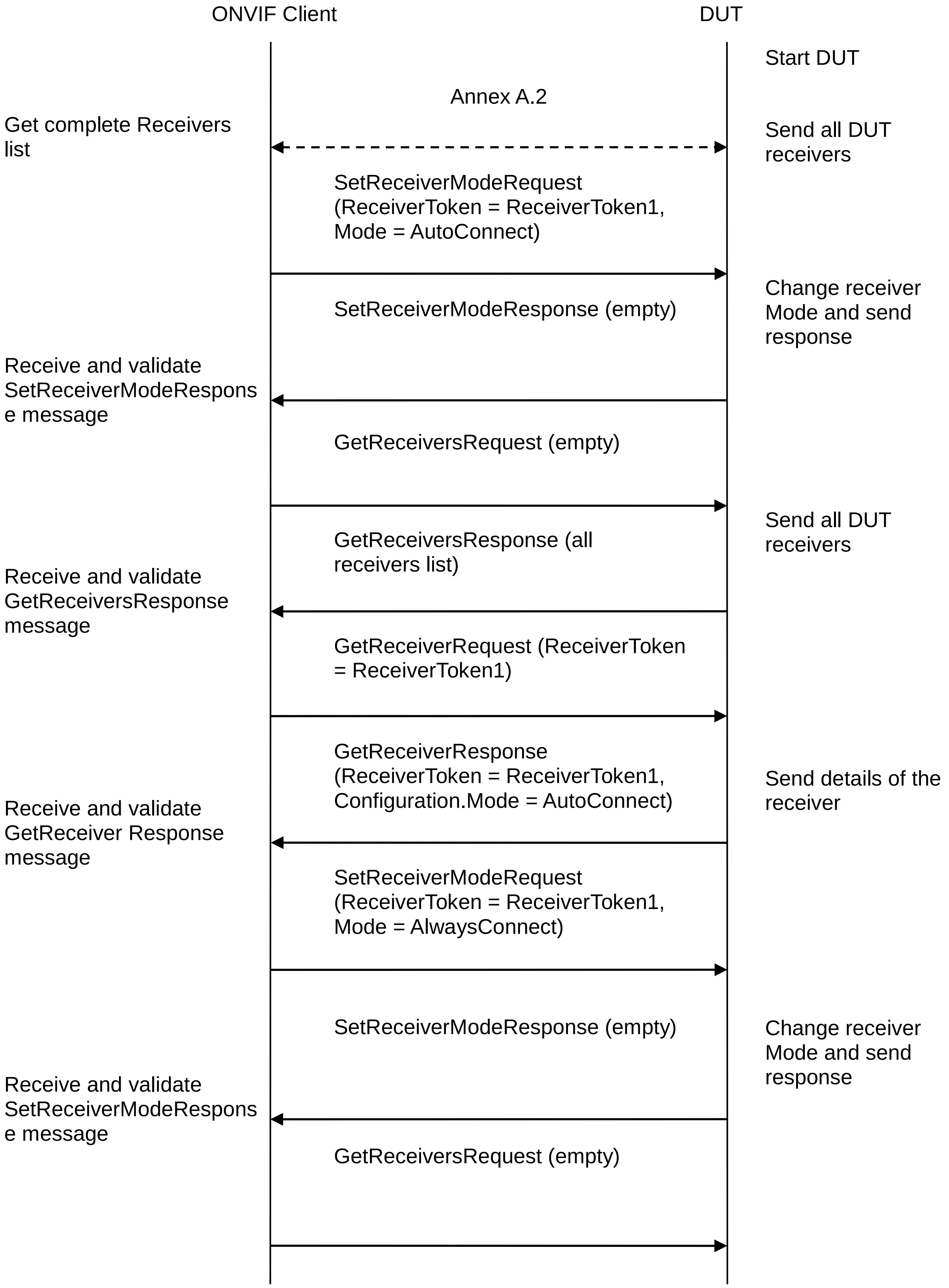
WSDL Reference: receiver.wsdl

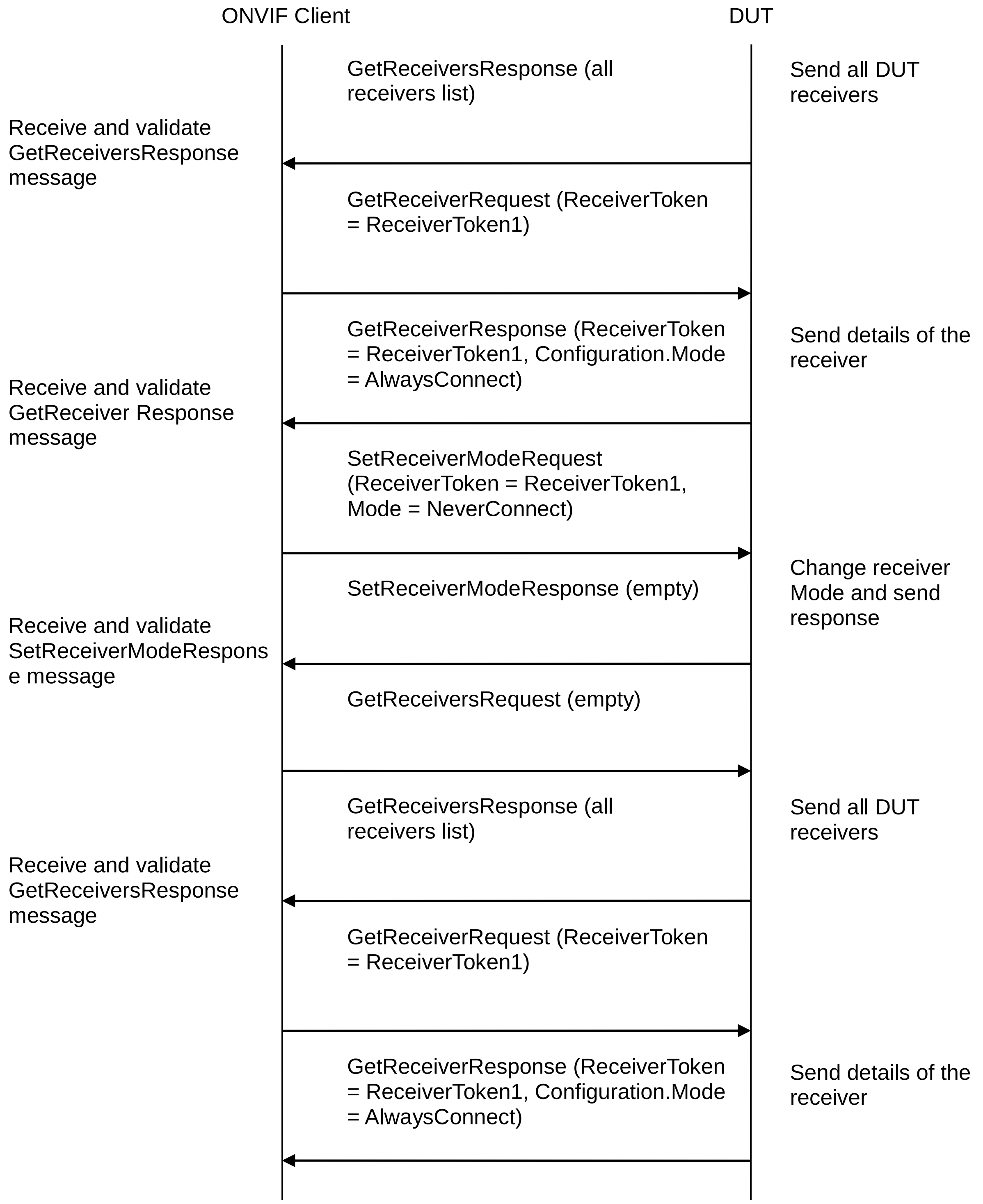
Test Purpose: To verify Receiver Mode change.

Pre-Requisite: Receiver Service was received from the DUT. At least one Receiver exists on the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete Receivers list
Send all DUT receivers
SetReceiverModeRequest (ReceiverToken = ReceiverToken1, Mode = AutoConnect)
SetReceiverModeResponse (empty)
Change receiver Mode and send response
Receive and validate SetReceiverModeResponse message
GetReceiversRequest (empty)
GetReceiversResponse (all receivers list)
Send all DUT receivers
Receive and validate GetReceiversResponse message
GetReceiverRequest (ReceiverToken = ReceiverToken1)
GetReceiverResponse (ReceiverToken = ReceiverToken1, Configuration.Mode = AutoConnect)
Send details of the receiver
Receive and validate GetReceiver Response message
SetReceiverModeRequest (ReceiverToken = ReceiverToken1, Mode = AlwaysConnect)
SetReceiverModeResponse (empty)
Change receiver Mode and send response
Receive and validate SetReceiverModeResponse message
GetReceiversRequest (empty)


ONVIF Client
DUT
GetReceiversResponse (all receivers list)
Send all DUT receivers
Receive and validate GetReceiversResponse message
GetReceiverRequest (ReceiverToken = ReceiverToken1)
GetReceiverResponse (ReceiverToken = ReceiverToken1, Configuration.Mode = AlwaysConnect)
Send details of the receiver
Receive and validate GetReceiver Response message
SetReceiverModeRequest (ReceiverToken = ReceiverToken1, Mode = NeverConnect)
SetReceiverModeResponse (empty)
Change receiver Mode and send response
Receive and validate SetReceiverModeRespons e message
GetReceiversRequest (empty)
GetReceiversResponse (all receivers list)
Send all DUT receivers
Receive and validate GetReceiversResponse message
GetReceiverRequest (ReceiverToken = ReceiverToken1)
GetReceiverResponse (ReceiverToken = ReceiverToken1, Configuration.Mode = AlwaysConnect)
Send details of the receiver

Test Procedure:

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

3. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).

4. ONVIF Client will invoke SetReceiverModeRequest message **(ReceiverToken = ReceiverToken1, Mode = AutoConnect).**

5. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.

6. Verify the GetReceiversResponse message (receivers list with changed receiver) from the DUT. Check that Receiver was changed with the specified mode.

7. ONVIF Client will invoke GetReceiverRequest message (Token = token1) to retrieve receiver configuration.

8. Verify the GetReceiverResponse message (Token = token1, Configuration.Mode = “AutoConnect”) from the DUT. Check that Receiver was changed with the specified mode.

9. ONVIF Client will invoke SetReceiverModeRequest message **(ReceiverToken = ReceiverToken1, Mode = AlwaysConnect).**

10. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.

11. Verify the GetReceiversResponse message (receivers list with changed receiver) from the DUT. Check that Receiver was changed with the specified mode.

12. ONVIF Client will invoke GetReceiverRequest message (Token = token1) to retrieve receiver configuration.

13. Verify the GetReceiverResponse message (Token = token1, Configuration.Mode = “AlwaysConnect”) from the DUT. Check that Receiver was changed with the specified mode.

14. ONVIF Client will invoke SetReceiverModeRequest message **(ReceiverToken = ReceiverToken1, Mode = NeverConnect).**

15. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.

16. Verify the GetReceiversResponse message (receivers list with changed receiver) from the DUT. Check that Receiver was changed with the specified mode.

17. ONVIF Client will invoke GetReceiverRequest message (Token = token1) to retrieve receiver configuration.

18. Verify the GetReceiverResponse message (Token = token1, Configuration.Mode = “NeverConnect”) from the DUT. Check that Receiver was changed with the specified mode.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid SetReceiverModeResponse message.

The DUT did not send a valid GetReceiverResponse message.

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT returned Receiver mode in GetReceiversResponse message that differ from the specified during Receiver mode change.

The DUT returned Receiver mode in GetReceiverResponse message that differ from the specified during Receiver mode change.

4.2.12 SET RECEIVER MODE – PERSISTENCE

Test Label: Set Receiver Mode Persistence Verification.

Test Case ID: RECEIVER-2-1-12

ONVIF Core Specification Coverage: Persistence (ONVIF Receiver Service Specification), SetReceiverMode (ONVIF Receiver Service Specification)

Command under test: SetReceiverMode

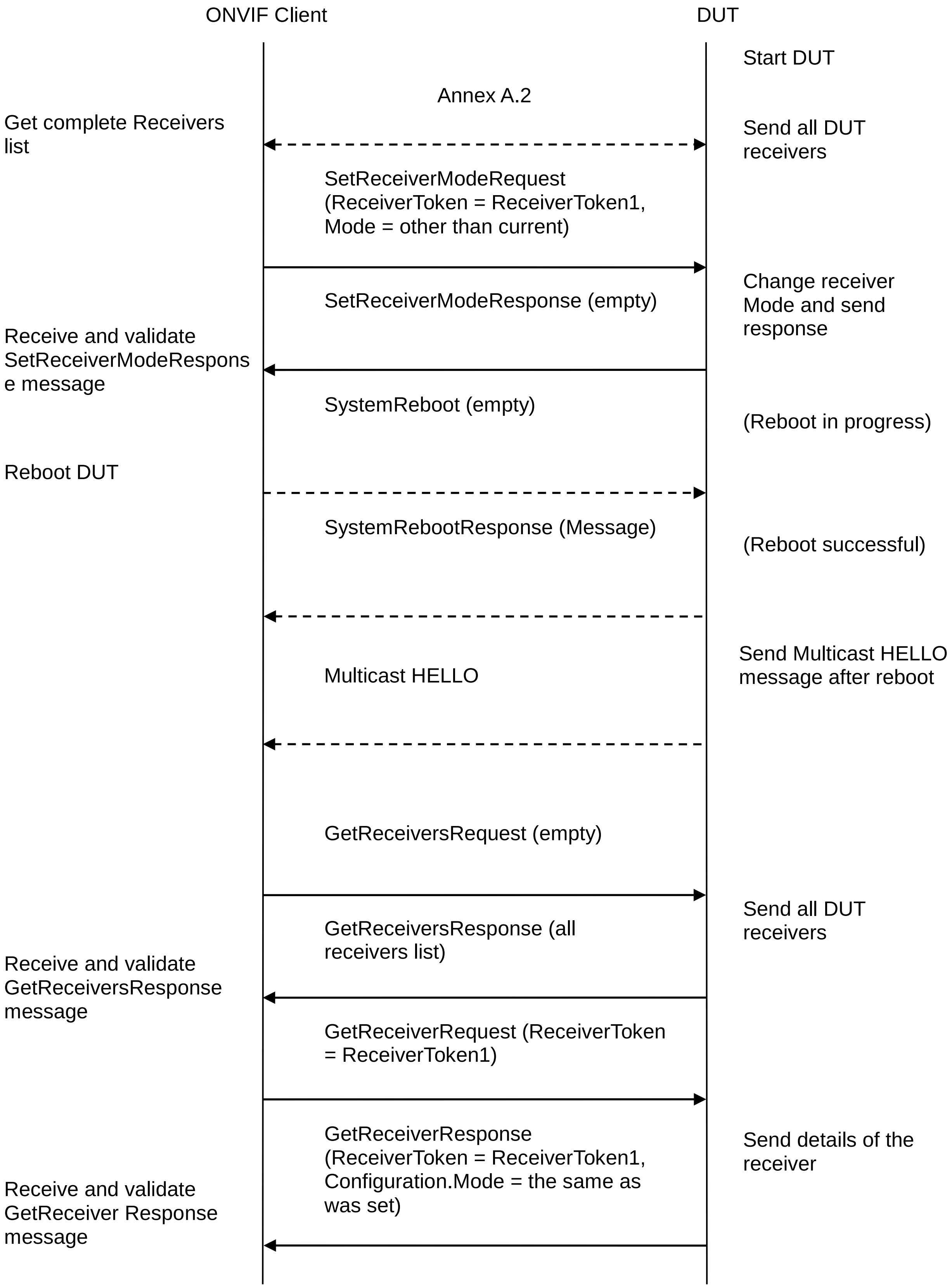
WSDL Reference: receiver.wsdl

Test Purpose: To verify Receiver Mode change persistence.

Pre-Requisite: Receiver Service was received from the DUT. At least one Receiver exists on the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete Receivers list
Send all DUT receivers
SetReceiverModeRequest (ReceiverToken = ReceiverToken1, Mode = other than current)
Change receiver Mode and send response
SetReceiverModeResponse (empty)
Receive and validate SetReceiverModeRespons e message
SystemReboot (empty)
(Reboot in progress)
Reboot DUT
SystemRebootResponse (Message)
(Reboot successful)
Multicast HELLO
Send Multicast HELLO message after reboot
GetReceiversRequest (empty)
Send all DUT receivers
GetReceiversResponse (all receivers list)
Receive and validate GetReceiversResponse message
GetReceiverRequest (ReceiverToken = ReceiverToken1)
GetReceiverResponse (ReceiverToken = ReceiverToken1, Configuration.Mode = the same as was set)
Send details of the receiver
Receive and validate GetReceiver Response message

Test Procedure:

1. Start an ONVIF Client.

2. Start the DUT.

3. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).

4. ONVIF Client will invoke SetReceiverModeRequest message **(ReceiverToken = ReceiverToken1, Mode = other than current).**

5. ONVIF Client will invoke SystemReboot message to reset the DUT.

6. Verify that DUT sends SystemRebootResponse message (example message string = “Rebooting in x seconds”).

7. DUT will send Multicast HELLO message after it is successfully rebooted.

8. ONVIF Client will verify the HELLO message sent by DUT.

9. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.

10. Verify the GetReceiversResponse message (receivers list with changed receiver) from the DUT. Check that Receiver was changed with the specified mode.

11. ONVIF Client will invoke GetReceiverRequest message (Token = token1) to retrieve receiver configuration.

12. Verify the GetReceiverResponse message (Token = token1, updated Configuration) from the DUT. Check that Receiver was changed with the specified mode.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid SetReceiverModeResponse message.

The DUT did not send a valid GetReceiverResponse message.

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT returned Receiver mode in GetReceiversResponse message that differ from the specified during Receiver mode change.

The DUT returned Receiver mode in GetReceiverResponse message that differ from the specified during Receiver mode change.

The DUT did not send SystemRebootResponse message.

The DUT did not send HELLO message.

4.2.13 SET RECEIVER MODE WITH INVALID TOKEN

Test Label: Set Receiver Mode Verification with Invalid Token.

Test Case ID: RECEIVER-2-1-13

ONVIF Core Specification Coverage: SetReceiverMode (ONVIF Receiver Service Specification)

Command under test: SetReceiverMode

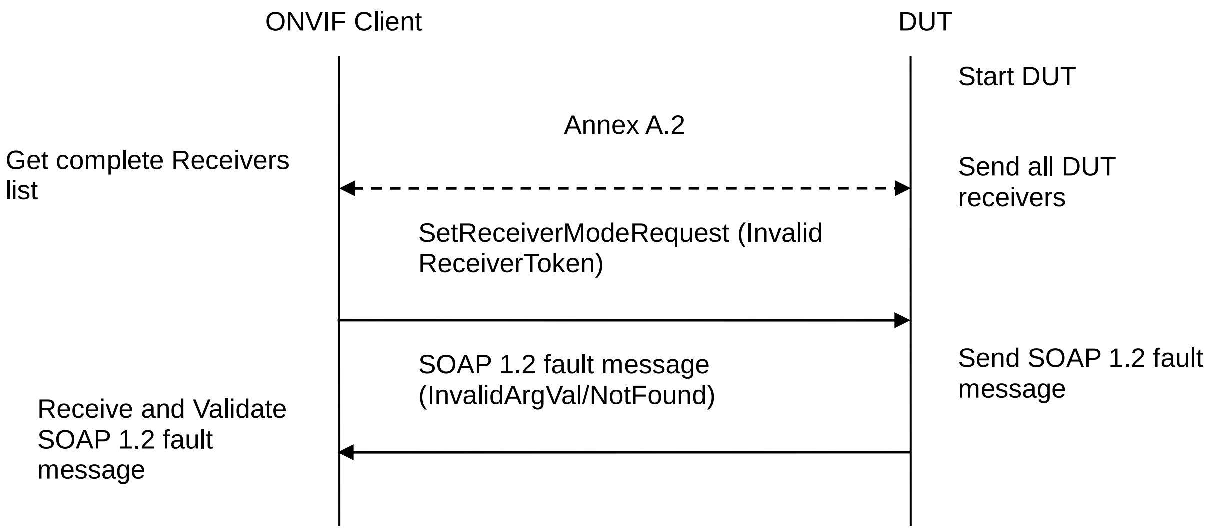
WSDL Reference: receiver.wsdl

Test Purpose: To verify Set Receiver Mode with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete Receivers list
Send all DUT receivers
SetReceiverModeRequest (Invalid ReceiverToken)
SOAP 1.2 fault message (InvalidArgVal/NotFound)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).
4. ONVIF Client will invoke SetReceiverModeRequest message **(invalid ReceiverToken**).
5. The DUT will generate SOAP 1.2 fault message (**InvalidArgVal/UnknownToken**).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send SOAP 1.2 fault message.

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

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

4.2.14 CONFIGURE RECEIVER – (RTP-Unicast/UDP)

Test Label: Get Receivers Verification.

Test Case ID: RECEIVER-2-1-16

ONVIF Core Specification Coverage: ConfigureReceiver, GetReceivers (ONVIF Receiver Service Specification)

Command under test: ConfigureReceiver, SetReceiverMode, GetReceiver

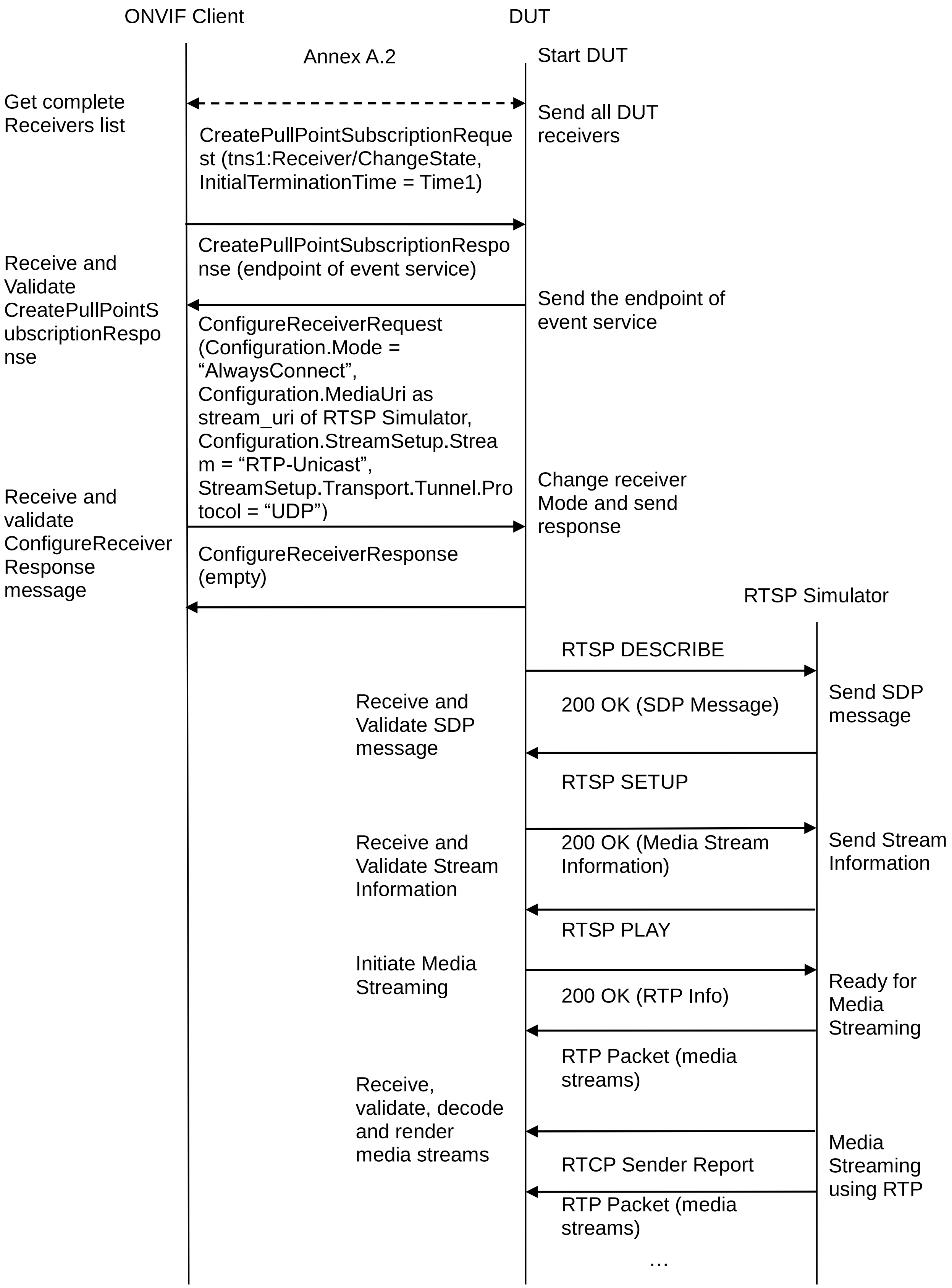
WSDL Reference: receiver.wsdl

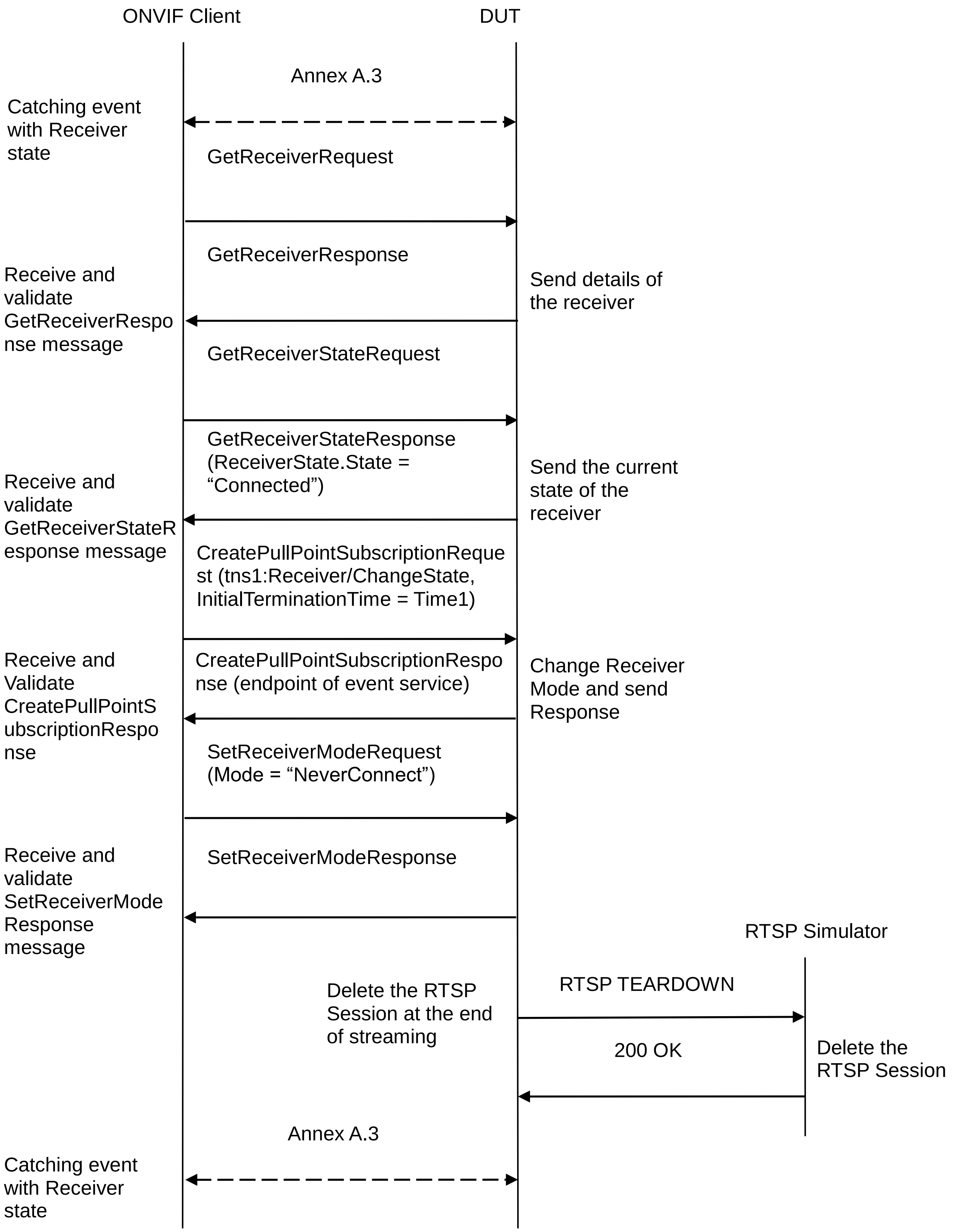
Test Purpose: To check receiver functionality.

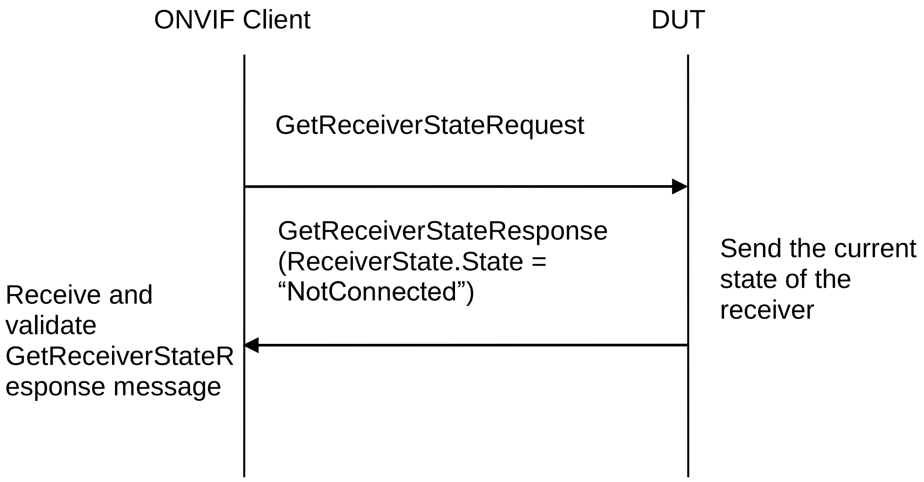
Pre-Requisite: Receiver Service was received from the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Annex A.2
Start DUT
Get complete Receivers list
Send all DUT receivers
CreatePullPointSubscriptionRequest (tns1:Receiver/ChangeState, InitialTerminationTime = Time1)
CreatePullPointSubscriptionResponse (endpoint of event service)
Receive and Validate CreatePullPointSubscriptionResponse
Send the endpoint of event service
ConfigureReceiverRequest (Configuration.Mode = “AlwaysConnect”, Configuration.MediaUri as stream_uri of RTSP Simulator, Configuration.StreamSetup.Stream = “RTP-Unicast”, StreamSetup.Transport.Tunnel.Protocol = “UDP”)
Change receiver Mode and send response
Receive and validate ConfigureReceiver Response message
ConfigureReceiverResponse (empty)
RTSP Simulator
RTSP DESCRIBE
Receive and Validate SDP message
200 OK (SDP Message)
Send SDP message
RTSP SETUP
Receive and Validate Stream Information
200 OK (Media Stream Information)
Send Stream Information
RTSP PLAY
Initiate Media Streaming
200 OK (RTP Info)
Ready for Media Streaming
RTP Packet (media streams)
Receive, validate, decode and render media streams
RTCP Sender Report
Media Streaming using RTP
RTP Packet (media streams)
…


ONVIF Client
DUT
Annex A.3
Catching event with Receiver state
GetReceiverRequest
GetReceiverResponse
Receive and validate GetReceiverResponse message
Send details of the receiver
GetReceiverStateRequest
GetReceiverStateResponse (ReceiverState.State = “Connected”)
Receive and validate GetReceiverStateResponse message
Send the current state of the receiver
CreatePullPointSubscriptionRequest (tns1:Receiver/ChangeState, InitialTerminationTime = Time1)
CreatePullPointSubscriptionResponse (endpoint of event service)
Receive and Validate CreatePullPointSubscriptionResponse
Change Receiver Mode and send Response
SetReceiverModeRequest (Mode = “NeverConnect”)
SetReceiverModeResponse
Receive and validate SetReceiverModeResponse message
RTSP Simulator
RTSP TEARDOWN
Delete the RTSP Session at the end of streaming
200 OK
Delete the RTSP Session
Annex A.3
Catching event with Receiver state


ONVIF Client
DUT
GetReceiverStateRequest
GetReceiverStateResponse (ReceiverState.State = “NotConnected”)
Send the current state of the receiver
Receive and validate GetReceiverStateResponse message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).
4. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Receiver/ChangeState Topic as Filter and an InitialTerminationTime = Time1 to check Receiver state.
5. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
6. ONVIF Client will invoke ConfigureReceiverRequest message (ReceiverToken as Token of the first Receiver in the GetReceiversResponse message, Configuration.Mode = “AlwaysConnect”, Configuration.MediaUri as stream_uri of RTSP Simulator, Configuration.StreamSetup.Stream = “RTP-Unicast”, StreamSetup.Transport.Tunnel.Protocol = “UDP”, no StreamSetup.Transport.Tunnel.Tunnel) to configure the receiver to receive media from RTSP Simulator.
7. Verify ConfigureReceiverResponse message from the DUT.
8. DUT invokes RTSP DESCRIBE request.
9. The RTSP SIMULATOR sends 200 OK message and SDP information.
10. DUT invokes RTSP SETUP request with transport parameter as RTP/UDP.
11. The RTSP SIMULATOR sends 200 OK message and the media stream information.
12. DUT invokes RTSP PLAY request.
13. The RTSP SIMULATOR sends 200 OK message and starts media streaming.
14. The RTSP SIMULATOR sends JPEG RTP media stream to DUT over UDP.
15. The RTSP SIMULATOR sends RTCP sender report to DUT.
16. DUT processed the received RTP and RTCP packets.

17. Execute Annex A.3 for catching event with current state of the receiver.

18. Verify that there is Notification Message with Source.SimpleItemDescription Description item with Name = "ReceiverToken" and Value is equal to "ReceiverToken1", Data.SimpleItemDescription Description item with Name = "New State" and Value is equal to "Connected".

19. ONVIF Client will invoke GetReceiverRequest message with ReceiverToken = ReceiverToken1.

20. Verify GetReceiverResponse message from the DUT. Check that GetReceiverResponse message contains the same parameters values as were changed in ConfigureReceiverRequest message.

21. ONVIF Client will invoke GetReceiverStateRequest message to check Receiver State.

22. Verify GetReceiverStateResponse message from the DUT. Check that ReceiverState.State = "Connected".

23. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Receiver/ChangeState Topic as Filter and an InitialTerminationTime = Time1 to check Receiver state.

24. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

25. ONVIF Client will invoke SetReceiverModeRequest message (ReceiverToken = "ReceiverToken1", Mode = "NeverConnect") to stop recording.

26. Verify SetReceiverModeResponse message from the DUT.

27. DUT invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.

28. The RTSP SIMULATOR sends 200 OK Response and terminates the RTSP Session.

29. Execute Annex A.3 for catching event with current state of the receiver.

30. Verify that there is Notification Message with Source.SimpleItemDescription Description item with Name = "ReceiverToken" and Value is equal to "ReceiverToken1", Data.SimpleItemDescription Description item with Name = "New State" and Value is equal to "NotConnected".

31. ONVIF Client will invoke GetReceiverStateRequest message to check Receiver State.

32. Verify GetReceiverStateResponse message from the DUT. Check that ReceiverState.State = "NotConnected".

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send a valid CreatePullPointSubscriptionResponse message.

The DUT did not send a valid ConfigureReceiverResponse message.

The DUT did not send a valid GetReceiverResponse message.

The DUT did not send a valid GetReceiverStateResponse message.

The DUT did not send a valid SetReceiverModeResponse message.

The DUT sent GetReceiverResponse message with parameters values differ from sent in ConfigureReceiverRequest message.

The DUT did not send Notify message with Source.SimpleItemDescription Description item with Name = "ReceiverToken” and Value is equal to “ReceiverToken1”, Data.SimpleItemDescription Description item with Name = "New State" and Value is equal to “Connected”.

The DUT did not send Notify message with Source.SimpleItemDescription Description item with Name = "ReceiverToken” and Value is equal to “ReceiverToken1”, Data.SimpleItemDescription Description item with Name = "New State" and Value is equal to “NotConnected”.

The DUT sent GetReceiverStateResponse with ReceiverState.State not equal to “Connected” in the step 22.

The DUT sent GetReceiverStateResponse with ReceiverState.State not equal to “NotConnected” in the step 32.

4.2.15 CONFIGURE RECEIVER – (RTP/RTSP/TCP)

Test Label: Get Receivers Verification.

Test Case ID: RECEIVER-2-1-17

ONVIF Core Specification Coverage: ConfigureReceiver, GetReceivers (ONVIF Receiver Service Specification)

Command under test: ConfigureReceiver, SetReceiverMode, GetReceiver

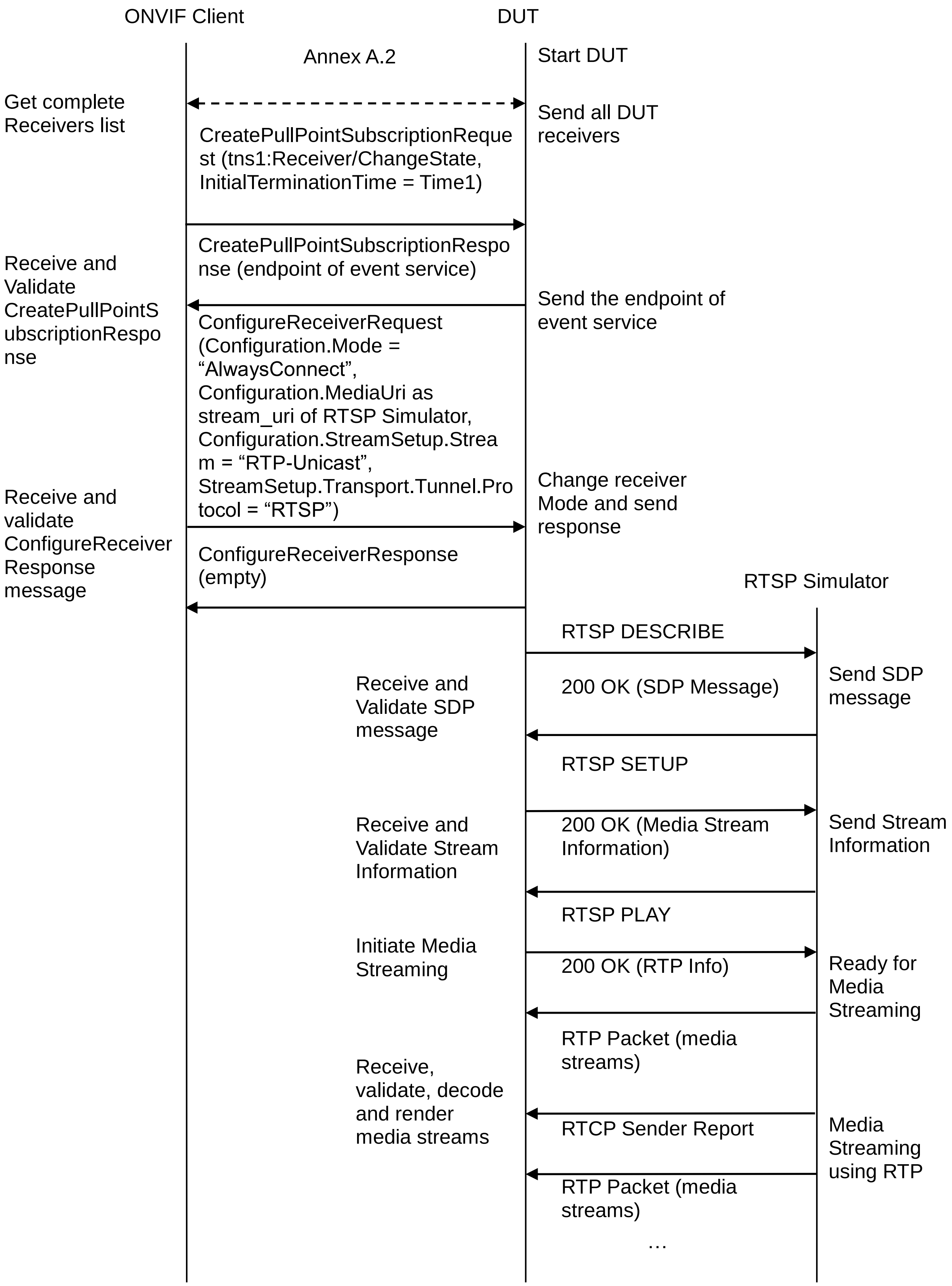
WSDL Reference: receiver.wsdl

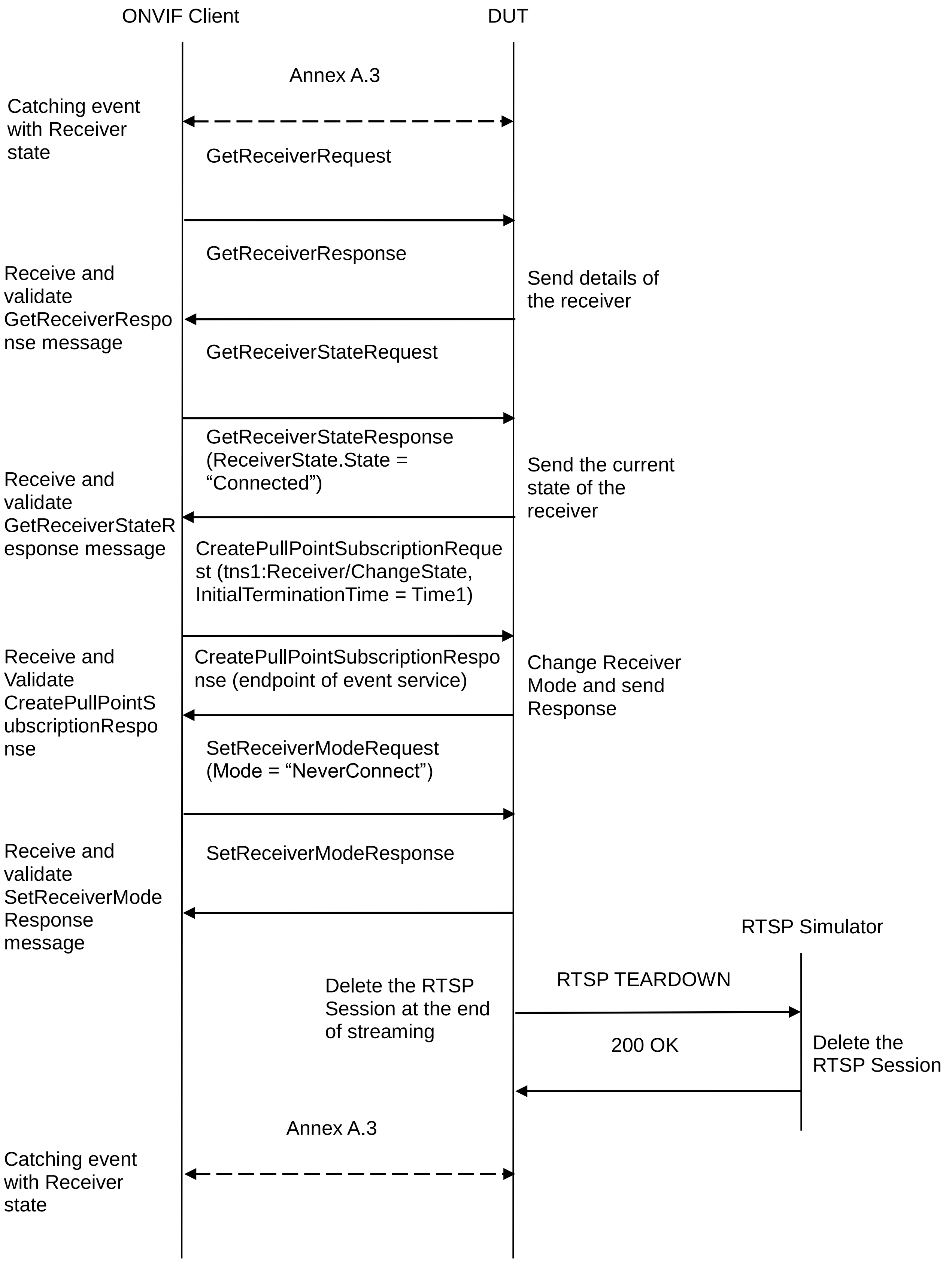
Test Purpose: To check receiver functionality.

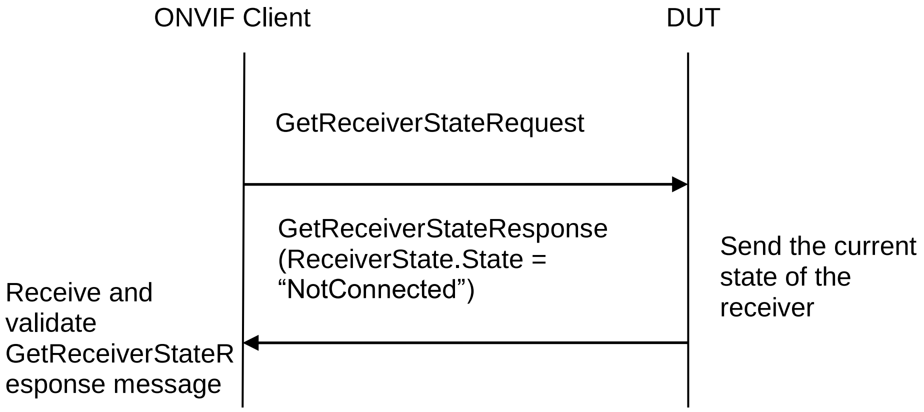
Pre-Requisite: Receiver Service was received from the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Annex A.2
Start DUT
Get complete Receivers list
Send all DUT receivers
CreatePullPointSubscriptionRequest (tns1:Receiver/ChangeState, InitialTerminationTime = Time1)
CreatePullPointSubscriptionResponse (endpoint of event service)
Receive and Validate CreatePullPointSubscriptionResponse
Send the endpoint of event service
ConfigureReceiverRequest (Configuration.Mode = “AlwaysConnect”, Configuration.MediaUri as stream_uri of RTSP Simulator, Configuration.StreamSetup.Stream = “RTP-Unicast”, StreamSetup.Transport.Tunnel.Protocol = “RTSP”)
Change receiver Mode and send response
Receive and validate ConfigureReceiver Response message
ConfigureReceiverResponse (empty)
RTSP Simulator
RTSP DESCRIBE
Receive and Validate SDP message
200 OK (SDP Message)
Send SDP message
RTSP SETUP
Receive and Validate Stream Information
200 OK (Media Stream Information)
Send Stream Information
RTSP PLAY
Initiate Media Streaming
200 OK (RTP Info)
Ready for Media Streaming
RTP Packet (media streams)
Receive, validate, decode and render media streams
RTCP Sender Report
Media Streaming using RTP
RTP Packet (media streams)
…


ONVIF Client
DUT
Annex A.3
Catching event with Receiver state
GetReceiverRequest
GetReceiverResponse
Receive and validate GetReceiverResponse message
Send details of the receiver
GetReceiverStateRequest
GetReceiverStateResponse (ReceiverState.State = “Connected”)
Receive and validate GetReceiverStateResponse message
Send the current state of the receiver
CreatePullPointSubscriptionRequest (tns1:Receiver/ChangeState, InitialTerminationTime = Time1)
CreatePullPointSubscriptionResponse (endpoint of event service)
Receive and Validate CreatePullPointSubscriptionResponse
Change Receiver Mode and send Response
SetReceiverModeRequest (Mode = “NeverConnect”)
SetReceiverModeResponse
Receive and validate SetReceiverModeResponse message
RTSP Simulator
RTSP TEARDOWN
Delete the RTSP Session at the end of streaming
200 OK
Delete the RTSP Session
Annex A.3
Catching event with Receiver state


ONVIF Client
DUT
GetReceiverStateRequest
GetReceiverStateResponse (ReceiverState.State = “NotConnected”)
Send the current state of the receiver
Receive and validate GetReceiverStateResponse message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).
4. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Receiver/ChangeState Topic as Filter and an InitialTerminationTime = Time1 to check Receiver state.
5. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
6. ONVIF Client will invoke ConfigureReceiverRequest message (ReceiverToken as Token of the first Receiver in the GetReceiversResponse message, Configuration.Mode = “AlwaysConnect”, Configuration.MediaUri as stream_uri of RTSP Simulator, Configuration.StreamSetup.Stream = “RTP-Unicast”, StreamSetup.Transport.Tunnel.Protocol = “RTSP”, no StreamSetup.Transport.Tunnel.Tunnel) to configure the receiver to receive media from RTSP Simulator.
7. Verify ConfigureReceiverResponse message from the DUT.
8. DUT invokes RTSP DESCRIBE request.
9. The RTSP SIMULATOR sends 200 OK message and SDP information.
10. DUT invokes RTSP SETUP request with transport parameter as RTP/TCP along with ‘interleaved’ parameter.
11. The RTSP SIMULATOR sends 200 OK message and the media stream information.
12. DUT invokes RTSP PLAY request.
13. The RTSP SIMULATOR sends 200 OK message and starts media streaming.
14. The RTSP SIMULATOR sends JPEG RTP media stream to DUT over UDP.
15. The RTSP SIMULATOR sends RTCP sender report to DUT.
16. DUT processed the received RTP and RTCP packets.

17. Execute Annex A.3 for catching event with current state of the receiver.

18. Verify that there is Notification Message with Source.SimpleItemDescription Description item with Name = "ReceiverToken” and Value is equal to “ReceiverToken1”, Data.SimpleItemDescription Description item with Name = "New State" and Value is equal to “Connected”.

19. ONVIF Client will invoke GetReceiverRequest message with ReceiverToken = ReceiverToken1.

20. Verify GetReceiverResponse message from the DUT. Check that GetReceiverResponse message contains the same parameters values as were changed in ConfigureReceiverRequest message.

21. ONVIF Client will invoke GetReceiverStateRequest message to check Receiver State.

22. Verify GetReceiverStateResponse message from the DUT. Check that ReceiverState.State = “Connected”.

23. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1:Receiver/ChangeState Topic as Filter and an InitialTerminationTime = Time1 to check Receiver state.

24. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.

25. ONVIF Client will invoke SetReceiverModeRequest message (ReceiverToken = “ReceiverToken1”, Mode = “NeverConnect”) to stop recording.

26. Verify SetReceiverModeResponse message from the DUT.

27. DUT invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.

28. The RTSP SIMULATOR sends 200 OK Response and terminates the RTSP Session.

29. Execute Annex A.3 for catching event with current state of the receiver.

30. Verify that there is Notification Message with Source.SimpleItemDescription Description item with Name = "ReceiverToken” and Value is equal to “ReceiverToken1”, Data.SimpleItemDescription Description item with Name = "New State" and Value is equal to “NotConnected”.

31. ONVIF Client will invoke GetReceiverStateRequest message to check Receiver State.

32. Verify GetReceiverStateResponse message from the DUT. Check that ReceiverState.State = “NotConnected”.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send a valid CreatePullPointSubscriptionResponse message.

The DUT did not send a valid ConfigureReceiverResponse message.

The DUT did not send a valid GetReceiverResponse message.

The DUT did not send a valid GetReceiverStateResponse message.

The DUT did not send valid SetReceiverModeResponse message.

The DUT sent GetReceiverResponse message with parameters values differ from sent in ConfigureReceiverRequest message.

The DUT did not send Notify message with Source.SimpleItemDescription Description item with Name = "ReceiverToken” and Value is equal to “ReceiverToken1”, Data.SimpleItemDescription Description item with Name = "New State" and Value is equal to “Connected”.

The DUT did not send Notify message with Source.SimpleItemDescription Description item with Name = "ReceiverToken” and Value is equal to “ReceiverToken1”, Data.SimpleItemDescription Description item with Name = "New State" and Value is equal to “NotConnected”.

The DUT sent GetReceiverStateResponse with ReceiverState.State not equal to “Connected” in the step 22.

The DUT sent GetReceiverStateResponse with ReceiverState.State not equal to “NotConnected” in the step 32.

4.2.16 CONFIGURE RECEIVER – INVALID MEDIA URI

Test Label: Get Receivers Verification.

Test Case ID: RECEIVER-2-1-18

ONVIF Core Specification Coverage: ConfigureReceiver, GetReceivers (ONVIF Receiver Service Specification)

Command under test: ConfigureReceiver, SetReceiverMode, GetReceiver

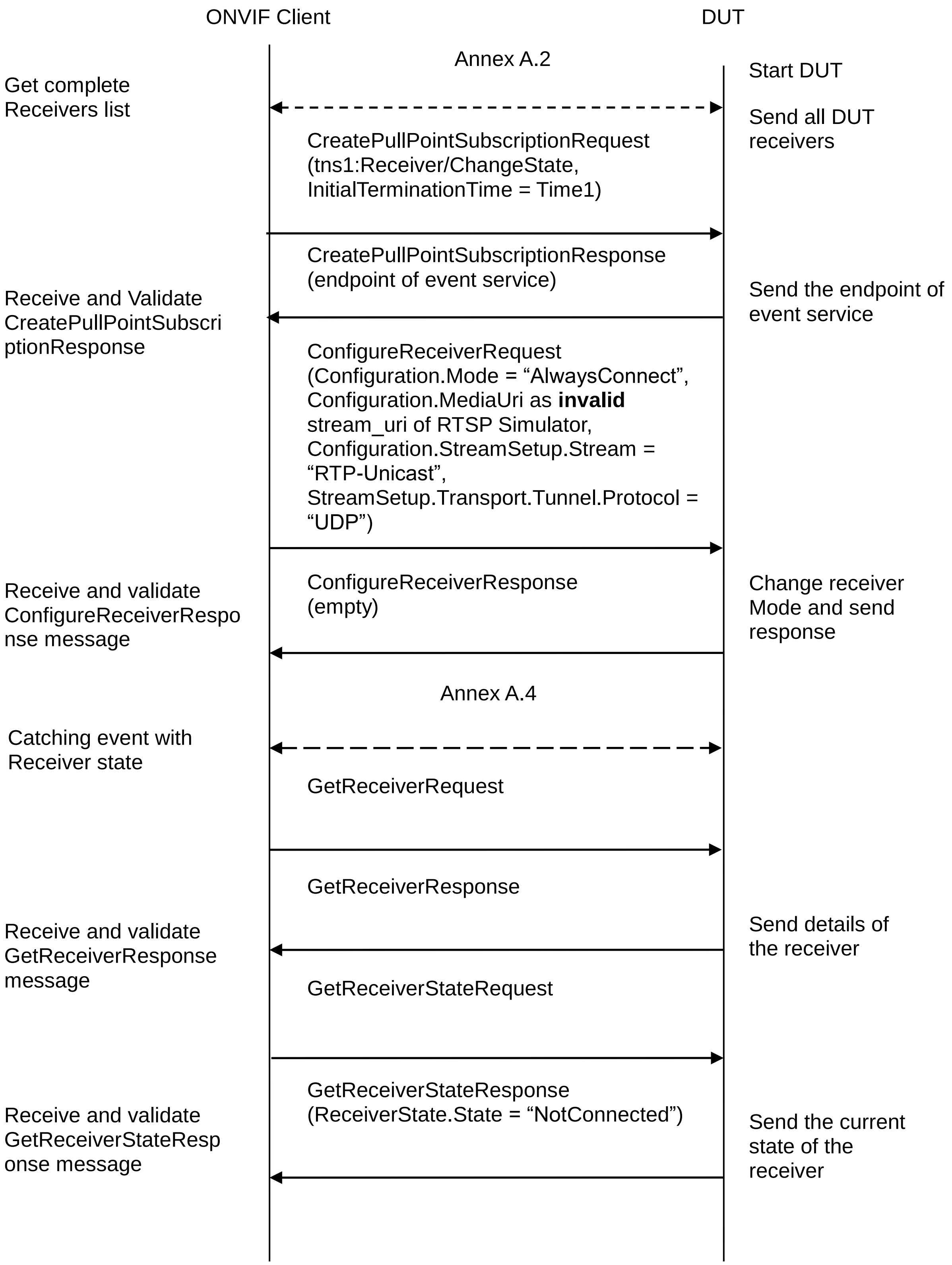
WSDL Reference: receiver.wsdl

Test Purpose: To check receiver functionality.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Annex A.2
Start DUT
Get complete Receivers list
Send all DUT receivers
CreatePullPointSubscriptionRequest (tns1:Receiver/ChangeState, InitialTerminationTime = Time1)
CreatePullPointSubscriptionResponse (endpoint of event service)
Receive and Validate CreatePullPointSubscriptionResponse
Send the endpoint of event service
ConfigureReceiverRequest (Configuration.Mode = “AlwaysConnect”, Configuration.MediaUri as **invalid** stream_uri of RTSP Simulator, Configuration.StreamSetup.Stream = “RTP-Unicast”, StreamSetup.Transport.Tunnel.Protocol = “UDP”)
ConfigureReceiverResponse (empty)
Receive and validate ConfigureReceiverResponse message
Change receiver Mode and send response
Annex A.4
Catching event with Receiver state
GetReceiverRequest
GetReceiverResponse
Receive and validate GetReceiverResponse message
Send details of the receiver
GetReceiverStateRequest
GetReceiverStateResponse (ReceiverState.State = “NotConnected”)
Receive and validate GetReceiverStateResponse message
Send the current state of the receiver

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).
4. ONVIF Client will invoke CreatePullPointSubscriptionRequest message with tns1: Receiver/ConnectionFailed Topic as Filter and an InitialTerminationTime = Time1 to check Receiver state.
5. Verify that the DUT sends a CreatePullPointSubscriptionResponse message.
6. ONVIF Client will invoke ConfigureReceiverRequest message (ReceiverToken as Token of the first Receiver in the GetReceiversResponse message, Configuration.Mode = “AlwaysConnect”, Configuration.MediaUri as invalid stream_uri of RTSP Simulator, Configuration.StreamSetup.Stream = “RTP-Unicast”, StreamSetup.Transport.Tunnel.Protocol = “UDP”, no StreamSetup.Transport.Tunnel.Tunnel) to configure the receiver to receive media from RTSP Simulator.
7. Verify ConfigureReceiverResponse message from the DUT.
8. Execute Annex A.4 for catching event with current state of the receiver.
9. Verify that there is Notification Message with TopicExpression = tns1:Receiver/ ConnectionFailed, Source.SimpleItemDescription Description item with Name = "ReceiverToken” and Value is equal to “ReceiverToken1”
10. ONVIF Client will invoke GetReceiverRequest message with ReceiverToken = ReceiverToken1.
11. Verify GetReceiverResponse message from the DUT. Check that GetReceiverResponse message contains the same parameters values as were changed in ConfigureReceiverRequest message.
12. ONVIF Client will invoke GetReceiverStateRequest message to check Receiver State.
13. Verify GetReceiverStateResponse message from the DUT. Check that ReceiverState.State = “NotConnected”.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send a valid CreatePullPointSubscriptionResponse message.

The DUT did not send a valid ConfigureReceiverResponse message.

The DUT did not send a valid GetReceiverResponse message.

The DUT did not send a valid GetReceiverStateResponse message.

The DUT sent GetReceiverResponse message with parameters values differ from sent in ConfigureReceiverRequest message.

The DUT did not send Notify message with TopicExpression = tns1:Receiver/ConnectionFailed and Source.SimpleItemDescription Description item with Name = "ReceiverToken" and Value is equal to "ReceiverToken1".

The DUT sent GetReceiverStateResponse with ReceiverState.State not equal to "NotConnected" at step 13.

4.2.17 CONFIGURE RECEIVER WITH INVALID TOKEN

Test Label: Configure Receiver Verification with Invalid Token.

Test Case ID: RECEIVER-2-1-19

ONVIF Core Specification Coverage: ConfigureReceiver (ONVIF Receiver Service Specification)

Command under test: ConfigureReceiver

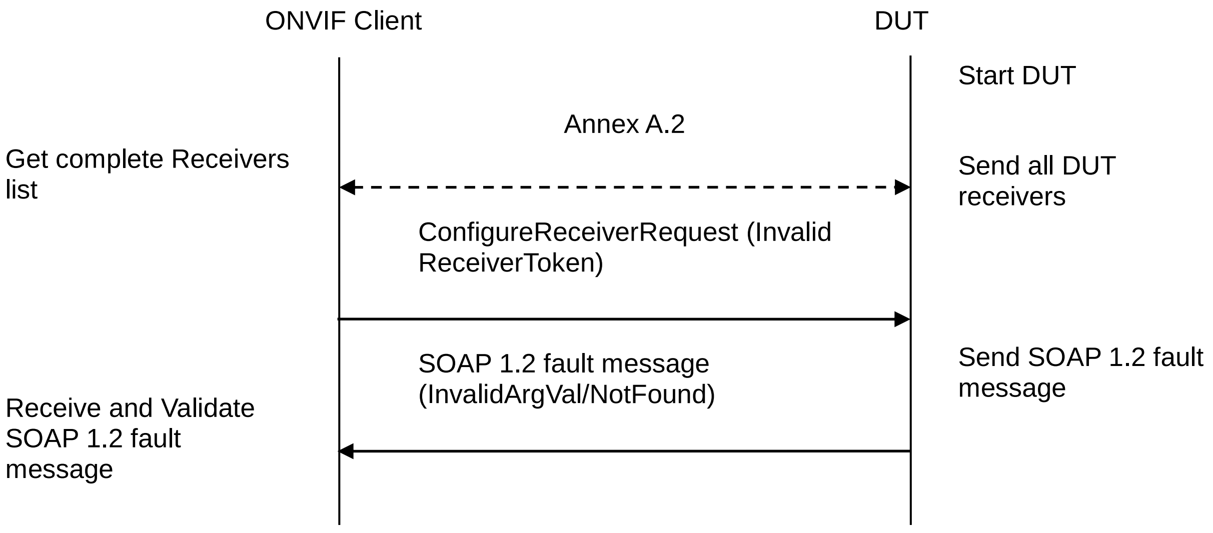
WSDL Reference: receiver.wsdl

Test Purpose: To verify Configure Receiver with invalid Token.

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

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete Receivers list
Send all DUT receivers
ConfigureReceiverRequest (Invalid ReceiverToken)
SOAP 1.2 fault message (InvalidArgVal/NotFound)
Send SOAP 1.2 fault message
Receive and Validate SOAP 1.2 fault message

Test Procedure:

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

3. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).
4. ONVIF Client will invoke ConfigureReceiverRequest message **(invalid ReceiverToken**).
5. The DUT will generate SOAP 1.2 fault message (**InvalidArgVal/UnknownToken**).

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT did not send SOAP 1.2 fault message.

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

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

4.2.18 CONFIGURE RECEIVER

Test Label: Configure Receiver Verification.

Test Case ID: RECEIVER-2-1-20

ONVIF Core Specification Coverage: GetReceivers (ONVIF Receiver Service Specification), GetReceiver (ONVIF Receiver Service Specification), ConfigureReceiver (ONVIF Receiver Service Specification)

Command under test: GetReceiver, ConfigureReceiver

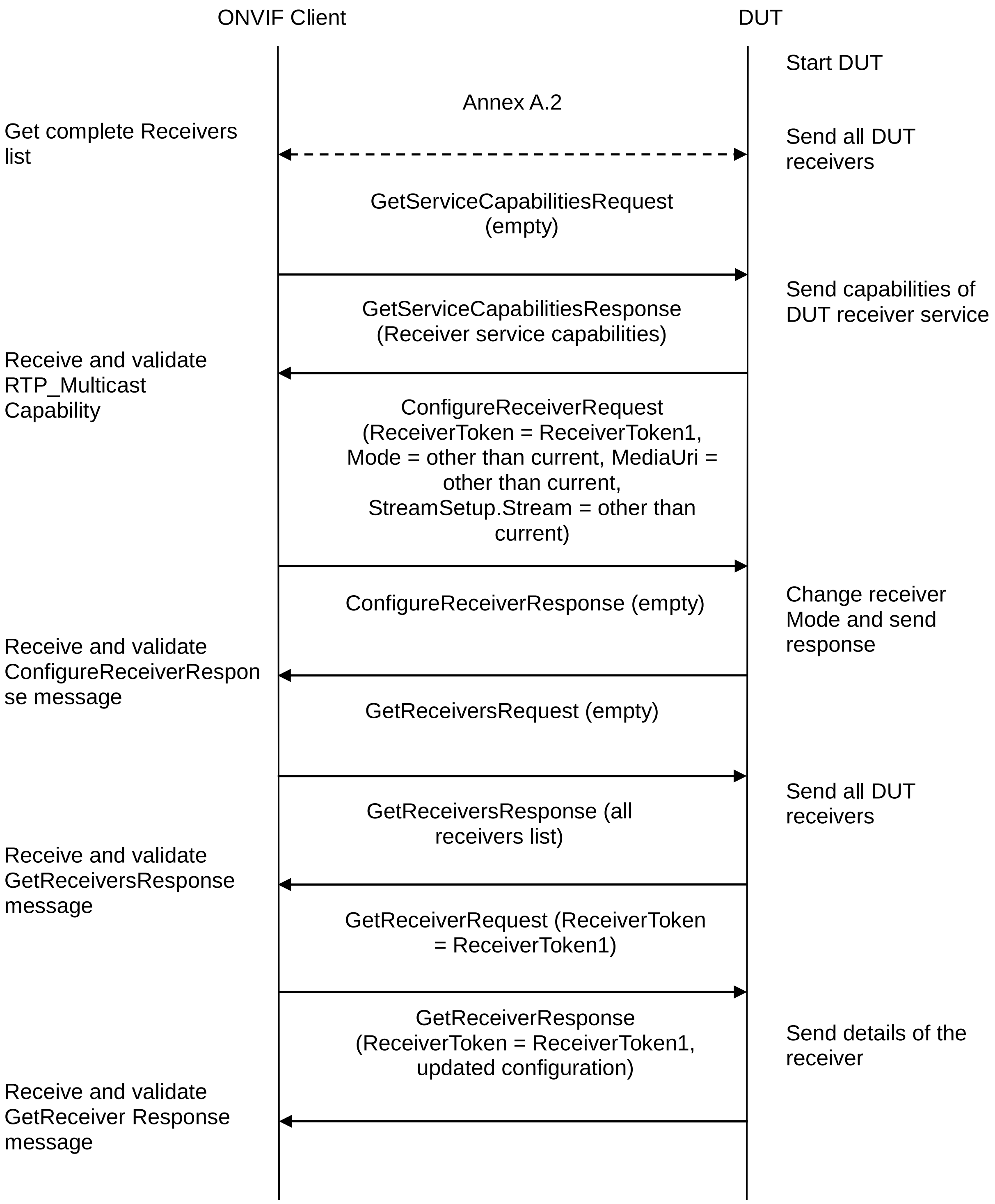
WSDL Reference: receiver.wsdl

Test Purpose: To verify Receiver configuration.

Pre-Requisite: Receiver Service was received from the DUT. At least one Receiver exists on the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete Receivers list
Send all DUT receivers
GetServiceCapabilitiesRequest (empty)
GetServiceCapabilitiesResponse (Receiver service capabilities)
Send capabilities of DUT receiver service
Receive and validate RTP_Multicast Capability
ConfigureReceiverRequest (ReceiverToken = ReceiverToken1, Mode = other than current, MediaUri = other than current, StreamSetup.Stream = other than current)
ConfigureReceiverResponse (empty)
Change receiver Mode and send response
Receive and validate ConfigureReceiverResponse message
GetReceiversRequest (empty)
Send all DUT receivers
GetReceiversResponse (all receivers list)
Receive and validate GetReceiversResponse message
GetReceiverRequest (ReceiverToken = ReceiverToken1)
GetReceiverResponse (ReceiverToken = ReceiverToken1, updated configuration)
Send details of the receiver
Receive and validate GetReceiver Response message

Test Procedure:

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

3. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).

4. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve the receiver service capabilities from the DUT.

5. Verify GetServiceCapabilitiesResponse message (Capabilities. RTP_Multicast).

6. ONVIF Client will invoke ConfigureReceiverRequest message **(ReceiverToken = ReceiverToken1, Mode = other than current, MediaUri = other than current, StreamSetup.Stream = other than current if DUT capability is allow).**

7. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.

8. Verify the GetReceiversResponse message (receivers list with changed receiver) from the DUT. Check that Receiver was changed with the specified configuration.

9. ONVIF Client will invoke GetReceiverRequest message (ReceiverToken = ReceiverToken1) to retrieve receiver configuration.

10. Verify the GetReceiverResponse message (ReceiverToken = ReceiverToken1, updated Configuration) from the DUT. Check that Receiver was changed with the specified configuration.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid ConfigureReceiverResponse message.

The DUT did not send a valid GetReceiverResponse message.

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT returned Receiver configuration in GetReceiversResponse message that differs from the specified during Receiver configuration change.

The DUT returned Receiver configuration in GetReceiverResponse message that differs from the specified during Receiver configuration change.

Note: MediaURI in the ConfigureReceiverRequest message shall be not greater than 28 octet length.

4.2.19 CONFIGURE RECEIVER – PERSISTENCE

Test Label: Configure Receiver Persistence Verification.

Test Case ID: RECEIVER-2-1-21

ONVIF Core Specification Coverage: Persistence (ONVIF Receiver Service Specification), ConfigureReceiver (ONVIF Receiver Service Specification)

Command under test: ConfigureReceiver

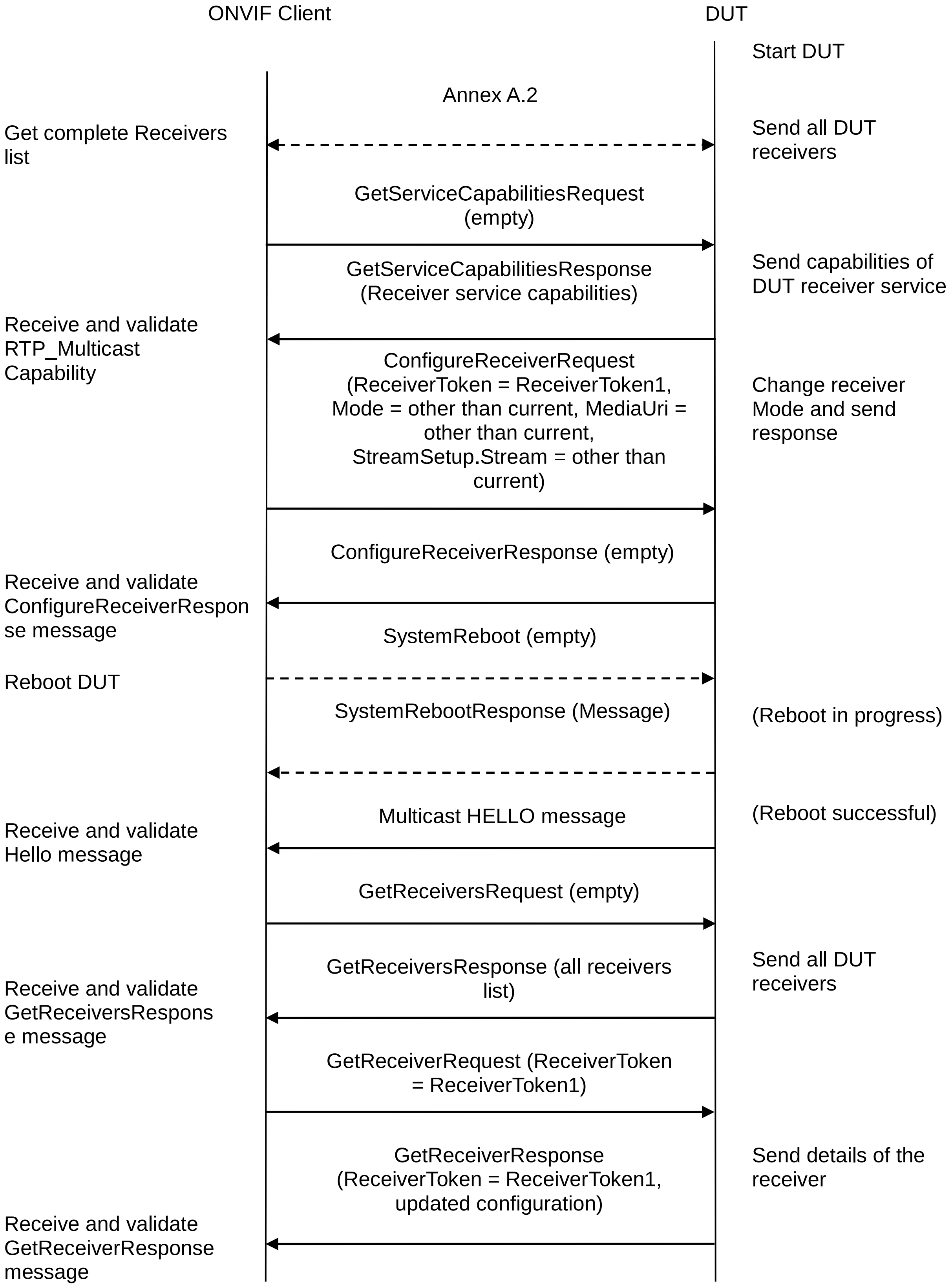
WSDL Reference: receiver.wsdl

Test Purpose: To verify Receiver configuration Persistence.

Pre-Requisite: Receiver Service was received from the DUT. At least one Receiver exists on the DUT.

Test Configuration: ONVIF Client and DUT

Test Sequence:


ONVIF Client
DUT
Start DUT
Annex A.2
Get complete Receivers list
Send all DUT receivers
GetServiceCapabilitiesRequest (empty)
GetServiceCapabilitiesResponse (Receiver service capabilities)
Send capabilities of DUT receiver service
Receive and validate RTP_Multicast Capability
ConfigureReceiverRequest (ReceiverToken = ReceiverToken1, Mode = other than current, MediaUri = other than current, StreamSetup.Stream = other than current)
Change receiver Mode and send response
ConfigureReceiverResponse (empty)
Receive and validate ConfigureReceiverResponse message
SystemReboot (empty)
Reboot DUT
SystemRebootResponse (Message)
(Reboot in progress)
Multicast HELLO message
(Reboot successful)
Receive and validate Hello message
GetReceiversRequest (empty)
GetReceiversResponse (all receivers list)
Send all DUT receivers
Receive and validate GetReceiversResponse message
GetReceiverRequest (ReceiverToken = ReceiverToken1)
GetReceiverResponse (ReceiverToken = ReceiverToken1, updated configuration)
Send details of the receiver
Receive and validate GetReceiverResponse message

Test Procedure:

1. Start an ONVIF Client.
2. Start the DUT.
3. ONVIF Client will retrieve a complete receivers list from the DUT and create a new Receiver if the list is empty (see Annex A.2).
4. ONVIF Client will invoke GetServiceCapabilitiesRequest message to retrieve the receiver service capabilities from the DUT.
5. Verify GetServiceCapabilitiesResponse message (Capabilities. RTP_Multicast).
6. ONVIF Client will invoke ConfigureReceiverRequest message (ReceiverToken = ReceiverToken1, Mode = other than current, MediaUri = other than current, StreamSetup.Stream = other than current if DUT capability is allow).
7. ONVIF Client will invoke SystemReboot message to reset the DUT.
8. Verify that the DUT sends SystemRebootResponse message (example message string = “Rebooting in x seconds”).
9. The DUT will send Multicast HELLO message after it is successfully rebooted.
10. ONVIF Client will verify the HELLO message sent by the DUT.
11. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.
12. Verify the GetReceiversResponse message (receivers list with changed receiver) from the DUT. Check that Receiver was changed with the specified configuration.
13. ONVIF Client will invoke GetReceiverRequest message (ReceiverToken = ReceiverToken1) to retrieve receiver configuration.
14. Verify the GetReceiverResponse message (ReceiverToken = ReceiverToken1, updated Configuration) from the DUT. Check that Receiver was changed with the specified configuration.

Test Result:

PASS –

The DUT passed all assertions.

FAIL –

The DUT did not send a valid ConfigureReceiverResponse message.

The DUT did not send a valid GetReceiverResponse message.

The DUT did not send a valid GetReceiversResponse message.

The DUT did not send a valid CreateReceiverResponse message.

The DUT returned Receiver configuration in GetReceiversResponse message that differs from the specified during Receiver configuration change.

The DUT returned Receiver configuration in GetReceiverResponse message that differs from the specified during Receiver configuration change.

The DUT did not send SystemRebootResponse message.

The DUT did not send HELLO message.

Note: MediaURI in the ConfigureReceiverRequest message shall be not greater than 28 octet length.

Annex A

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

A.1 Comparison of Receivers in GetReceiverResponse and GetReceiversResponse messages

Receivers in GetReceiversResponse message and in GetReceiverResponse message will be assumed as different in the following cases:

1. Configuration.Mode element values are different.
2. Configuration.MediaUri element values are different.
3. Configuration.StreamSetup.Stream element values are different.
4. Configuration.StreamSetup.Transport.Protocol element values are different.
5. Configuration.StreamSetup.Transport.Tunnel element is skipped only for GetReceiversResponse message or only for GetReceiverResponse message.
6. Configuration.StreamSetup.Transport.Tunnel.Protocol element values are different.
7. Configuration.StreamSetup.Transport.Tunnel.[...].Tunnel (check recursively) element is skipped only for GetReceiversResponse message or only for GetReceiverResponse message.
8. Configuration.StreamSetup.Transport.Tunnel.[...].Tunnel.Protocol (check recursively) element values are different.

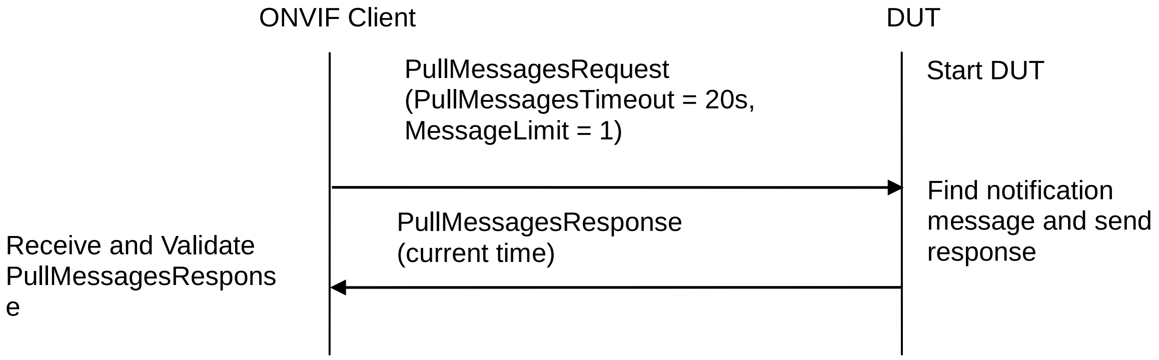
A.2 Get Complete Receivers List and Create a Receiver if the List is Empty

ONVIF Client follows the following procedure of Retrieving Receivers List and creation of a Receiver if the list is Empty:

1. ONVIF Client will invoke GetReceiversRequest message to retrieve a complete receivers list.
2. Verify that the GetReceiversResponse message contains at least one Receiver.
3. If there is at least one Receiver, then skip steps 4-5 and return to the Test Procedure.
4. ONVIF Client will invoke CreateReceiverRequest message (Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport.Protocol = ’UDP’) to create new Receiver.
5. Verify the CreateReceiverResponse message (ReceiverToken1, Configuration.Mode = “NeverConnect”, Configuration.MediaUri = validUri, Configuration.StreamSetup.Stream = ’RTP-Unicast’, Configuration.StreamSetup.Transport.Protocol = ’UDP’) from the DUT. Check that CreateReceiverResponse message contains the same parameters values as were sent in CreateReceiverRequest message.

A.3 PullMessages algorithm for check Receiver State changing

If the receiver changes state, the device sends the correspond event. Algorithm of PullMessages sending for catching event with expected state.


ONVIF Client
DUT
PullMessagesRequest (PullMessagesTimeout = 20s, MessageLimit = 1)
Start DUT
Find notification message and send response
PullMessagesResponse (current time)
Receive and Validate PullMessagesRespons e

1. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1 to find NotificationMessage containing event with required ReceiverState for used Recording.

2. Verify PullMessagesResponse message.

3. If no events are returned and CurrentTime of sending PullMessages is more than T1+delta, where delta is Operation delay time, skip other steps. If no events are returned and CurrentTime of sending PullMessages is less or equal than T1+delta go to the step 1.

4. If event is returned, check UTC Time of the received event. If UTC Time is more than T1+delta skip other steps.

5. Find event with Source.SimpleItemDescription Description item with Name = "ReceiverToken" and Value is equal to "ReceiverToken1", Data.SimpleItemDescription Description item with Name = "New State" and Value is equal to required receiver state.

6. If event is not found go to the step 1, otherwise, go to the next step.

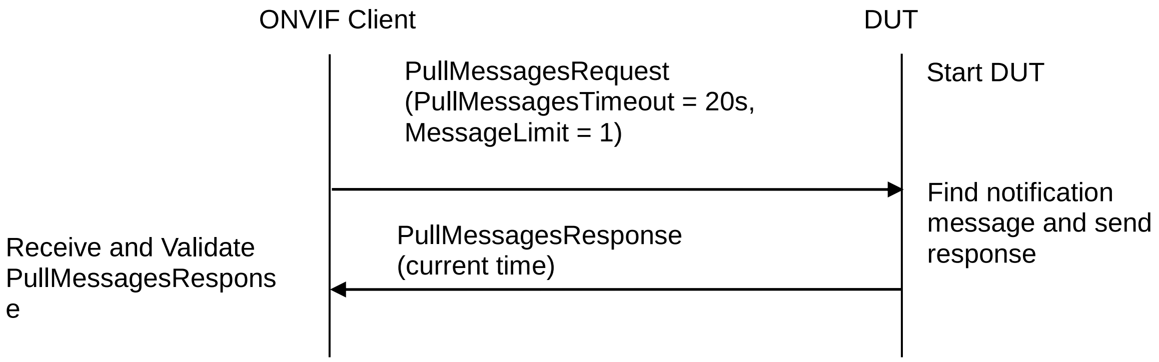
Test will be assumed as failed in the case:

The DUT did not send a valid PullMessagesResponse message.

The DUT did not send NotificationMessage with event which has Source.SimpleItemDescription Description item with Name = "ReceiverToken" and Value is equal to "ReceiverToken1", Data.SimpleItemDescription Description item with Name = "New State" and Value is equal to required receiver state.

A.4 PullMessages algorithm for Receiver Connection Failed event

If the receiver cannot connect to source, the device sends the correspond event. Algorithm of PullMessages sending for catching event with expected state.


ONVIF Client
DUT
PullMessagesRequest (PullMessagesTimeout = 20s, MessageLimit = 1)
Start DUT
Find notification message and send response
Receive and Validate PullMessagesRespons e
PullMessagesResponse (current time)

1. ONVIF Client will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 1 to find NotificationMessage containing event with required ReceiverState for used Recording.

2. Verify PullMessagesResponse message.

3. If no events are returned and CurrentTime of sending PullMessages is more than T1+delta, where delta is Operation delay time, skip other steps. If no events are returned and CurrentTime of sending PullMessages is less or equal than T1+delta go to the step 1.

4. If event is returned, check UTC Time of the received event. If UTC Time is more than T1+delta skip other steps.

5. Find event with Source.SimpleItemDescription Description item with Name = "ReceiverToken” and Value is equal to “ReceiverToken1”.

6. If event is not found, go to the step 1; otherwise go to the next step.

Test will be assumed as failed in the case:

The DUT did not send valid PullMessagesResponse message.

The DUT did not send NotificationMessage with event with topic tns1: Receiver/ConnectionFailed which has Source.SimpleItemDescription Description item with Name = "ReceiverToken” and Value is equal to “ReceiverToken1.