Driving IP-based physical security through global standardization

ONVIF™

Test Specification

Version 1.02.2

December, 2010

Driving IP-based physical security through global standardization

Driving IP-based physical security through global standardization

Revision History

Ver.	Date	Description
1.0	12th/Dec, 2008	First issue
1.01	16th/Sep, 2009	Incorporated submitted errata #1 - #26
1.02	10th/Aug, 2010	Extended coverage from previous version
1.02.2	16th/Dec, 2010	Extended coverage from previous version

Driving IP-based physical security through global standardization

Table of Contents

Contributors List

Version 1.0

Daniel Elvin	Axis Communications AB
Göran Haraldsson	Axis Communications AB
Markus Wierny	Bosch Security Systems
Rainer Bauereiss	Bosch Security Systems
Susanne Kinza	Bosch Security Systems
Andreas Schneider	Sony Corporation
Michio Hirai (Ed.)	Sony Corporation
Norio Ishibashi	Sony Corporation
Santosh Kulkarni	Sony Corporation

Version 1.01

Baldvin Gislason Bern	Axis Communications AB
Daniel Elvin	Axis Communications AB
Markus Wierny	Bosch Security Systems
Nguyen Dieu Thanh	Bosch Security Systems
Rainer Bauereiss	Bosch Security Systems
Susanne Kinza	Bosch Security Systems
Andreas Schneider	Sony Corporation
Balakrishna Vinturi	Sony Corporation
Michio Hirai (Ed.)	Sony Corporation
Santosh Kulkarni	Sony Corporation

Version 1.02

Baldvin Gislason Bern	Axis Communications AB
Bekim Berisha	Axis Communications AB
Susanne Kinza	Bosch Security Systems
Dharma Ramaiah	Canon Inc.
Takahiro Iwasaki	Canon Inc.
Takeshi Asahi	Hitachi, Ltd.
Hiroaki Ootake	Panasonic System Networks Co.,Ltd.
Hirokazu Kitaoka	Panasonic System Networks Co.,Ltd.
Tsuyoshi Ogata	Panasonic System Networks Co.,Ltd.
Balakrishna Vinturi	Sony Corporation
Michio Hirai (Ed.)	Sony Corporation

1 Introduction

The goal of the ONVIF test specification is to make it possible to realize fully interoperable network video implementations from different network video vendors. The ONVIF test specification describes the test cases needed to verify the [ONVIF Core] requirements. It also describes the test framework, test setup, pre-requisites, test policies needed for the execution of the described test cases.

This ONVIF Test Specification acts as a supplementary document to the [ONVIF Core], clarifying the requirements wherever needed. And also this specification acts an input document to the development of test tool which will be used to test the Network Video Transmitter (NVT) implementation conformance towards the [ONVIF Core]. This test tool is referred as Network Video Client (NVC) hereafter.

An NVT implementation which claims conformance to [ONVIF Core] MUST successfully execute all the mandatory test cases defined in this document.

1.1 Scope

This ONVIF Test Specification defines and regulates the conformance testing procedure for the ONVIF NVT implementation. 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 NVT implementation as per [ONVIF Core].

The principal intended purposes are:

1. Provide self-assessment tool for implementations.
2. Provide comprehensive test suite coverage for [ONVIF Core].

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. Protocol Implementation Conformance test for HTTPS, HTTP, RTP and RTSP protocols.
4. Poor streaming performance test (audio/video distortions, missing audio/video frames, incorrect lip synchronization etc)
5. Wi-Fi Conformance test

This ONVIF Test Specification will not cover the complete set of requirements as defined in [ONVIF Core]; instead it would cover subset of it. The scope of this specification is to cover all the mandatory requirements of [ONVIF Core] and some of the optional requirements.

This ONVIF Test Specification covers the following list of functional blocks in [ONVIF Core]. The following sections describe the brief over view and scope of each functional block.

1.1.1 IP Configuration

IP Configuration covers the test cases needed for the verification of IP configuration features as mentioned in [ONVIF Core]. IP configuration section defines the ONVIF IP configuration compliance requirements and recommendations.

The scope of this specification is to cover following configurations.

- IPv4 configuration.
  - Static IP configuration
  - Link-local address configuration.
  - DHCP configuration.
- IPv6 configuration.
  - Static IP configuration.
  - Stateless IP configuration.
  - Stateful IP configuration.

1.1.2 Device Discovery

Device discovery and location of the device services in the network are achieved using a multicast discovery protocol defined in WS-Discovery. The communication between client and target service is done using Web Services, notably SOAP/UDP.

Device Discovery testing tests the following:

- Device discovery in the ad-hoc network.
- Location of one or more device services.
- Enable discovery of service by type and within scope.
- SOAP 1.2 envelopes.
- SOAP 1.2 fault messages.

Refer Table 1 for Device Discovery Test.

Table 1 Device Discovery

Feature	Messages
Device Discovery	Hello Probe Probe Match Bye

1.1.3 Device Management

Device Management covers the test cases for the verification of the device service as mentioned in [ONVIF Core]. The device service is the entry point to all other services provided by a device.

The scope of this specification is to cover interfaces with regard to following subcategories of device service.

- Capabilities

- Network
- System
- Security

1.1.4 Media Configuration

Media Configuration section covers the test cases needed for the verification of media service features as mentioned in [ONVIF Core]. Media service is used to configure the media and other real time streaming configurations.

Briefly it covers the following things

- Manage media profiles.
- Manage configuration entities.
- Initiate and manipulate Audio/Video streams for different media formats.

The scope of this specification is to cover following configuration entities and Audio/Video media formats.

Configuration Entities:

- Video source configuration
- Audio source configuration
- Video encoder configuration
- Audio encoder configuration
- PTZ configuration
- Metadata configuration

Video Codec:

- JPEG QVGA
- MPEG-4, Simple Profile
- H.264, Baseline

Audio Codec:

- G.711
- G.726
- AAC

1.1.5 Real Time Streaming

Real Time Streaming covers the test cases needed for the verification of Real time streaming features as mentioned in [ONVIF Core]. Real time streaming section defines different media streaming options based on RTP for video, audio and metadata streams. Media control is done using RTSP protocol.

The scope of this specification to cover the following real time streaming options for JPEG, MPEG4 and H.264 video streams, and JPEG/ G.711, JPEG/ G.726 and JPEG/ AAC Audio & Video streams.

- RTSP control requests
- RTP unicast over UDP
- RTP over RTSP over TCP
- RTP over RTSP over HTTP over TCP
- RTCP

1.1.6 Event Handling

Event handling covers the test cases for the verification of the Event service as mentioned in [ONVIF Core] and [ONVIF Event WSDL].

The event handling test cases cover the following mandatory interfaces:

- **Basic Notification Interface**

  This test specification provides test cases to verify the implementation of the Basic Notification Interface of a device. The mechanisms to subscribe and unsubscribe to an event are covered as well as the mechanism to renew a subscription manager and receive events via Notify messages. The correct use of the message content filter is also tested.

- **Real Time Pull Point Notification Interface**

  Test cases to verify the implementation of the Realtime PullPoint Interface are provided by this test specification. The CreatePullPointSubscription command is tested as well as the PullMessages command in combination with message content filtering to retrieve the events.

- **Notification Streaming Interface**

  The test specification provides test cases to verify the Notification streaming interface. The correct implementation of the creation, deletion and modification of metadata configurations inside a profile are tested. Additionally the event streaming using RTSP and RTP is covered by this specification.

1.1.7 PTZ Control

PTZ Control covers the test cases for the verification of the PTZ service as mentioned in [ONVIF Core].

Refer Table 2 for PTZ Control Test.

Table 2 PTZ Control

Feature	Messages
PTZ Node	GetNodes GetNode

PTZ Configuration	GetConfigurations GetConfiguration GetConfigurationOptions SetConfiguration
Move Operations	AbsoluteMove RelativeMove ContinuousMove Stop GetStatus
Preset operations	SetPreset GetPresets GotoPreset RemovePreset
Home Position operations	GotoHomePosition SetHomePosition
Auxiliary operations	SendAuxiliaryCommand
Predefined PTZ spaces	AbsolutePositionSpaces RelativeTranslationSpaces ContinuousVelocitySpaces SpeedSpaces

1.1.8 Security

Security covers the test cases needed for the verification of required security features as mentioned in [ONVIF Core]. The scope of this specification is limited to Message level security and Username Token Profile.

1.2 Normative References

[ONVIF Core]	ONVIF Core Specification version 1.02 June, 2010.
[ONVIF DM WSDL]	ONVIF Device Management Service WSDL, ver 1.1, June 2010.
[ONVIF Event WSDL]	ONVIF Event Service WSDL, ver 1.1, June 2010.
[ONVIF Media WSDL]	ONVIF Media Service WSDL, ver 1.1, June 2010.
[ONVIF PTZ WSDL]	ONVIF PTZ Service WSDL, ver 2.0, June 2010.
[ONVIF Schema]	ONVIF Schema, ver 1.1, June 2010.
[ONVIF Topic Namespace]	ONVIF Topic Namespace XML, ver 1.1, June 2010.
[ONVIF Conformance]	ONVIF Conformance Process Specification, May 2009
[RFC 758]	"Assigned Numbers", J. Postel, August 1979 URL:http://www.ietf.org/rfc/rfc758
[RFC 952]	"DOD INTERNET HOST TABLE SPECIFICATION", K. Harrenstien, M. Stahl and E. Feinler, October 1985 URL:http://www.ietf.org/rfc/rfc952
[RFC 1123]	"Requirements for Internet Hosts -- Application and Support", R. Braden, October 1989 URL:http://www.ietf.org/rfc/rfc1123
[RFC 2119]	"Key words for use in RFCs to Indicate Requirement Levels". S. Bradner, March 1997. URL:http://www.ietf.org/rfc/rfc2119
[RFC 2131]	"Dynamic Host Configuration Protocol", R. Droms, March 1997. URL:http://www.ietf.org/rfc/rfc2131
[RFC 2136]	"Dynamic Updates in the Domain Name System (DNS UPDATE)", P. Vixie et. Al, April 1997. URL:http://www.ietf.org/rfc/rfc2136
[RFC 2326]	"Real Time Streaming Protocol (RTSP)", H. Schulzrinne, A. Rao and R. Lanphier, April 1998. URL:http://www.ietf.org/rfc/rfc2326
[RFC 2435]	"RFC2435 - RTP Payload Format for JPEG-compressed Video", L. Berc et al., October 1998. URL:http://www.ietf.org/rfc/rfc2435.txt
[RFC 2780]	"IANA Allocation Guidelines For Values in the Internet", S. Bradner and V. Paxson, March 2000 URL:http://www.ietf.org/rfc/rfc2780
[RFC 3315]	"Dynamic Host Configuration Protocol for IPv6 (DHCPv6)", R. Droms et al., July 2003. URL:http://www.ietf.org/rfc/rfc3315.txt
[RFC 3550]	"RTP: A Transport Protocol for Real-Time Applications", H. Schulzrinne et. Al., July 2003. URL:http://www.ietf.org/rfc/rfc3550
[RFC 3927]	"Dynamic Configuration of IPv4 Link-Local Addresses", S. Cheshire, B. Aboba and E. Guttman, May 2005. URL:http://www.ietf.org/rfc/rfc3927
[RFC 3984]	"RTP Payload Format for H.264 Video", S. Wenger et al., February 2005. URL:http://www.ietf.org/rfc/rfc3984
[RFC 3986]	"Uniform Resource Identifier (URI): Generic Syntax", T. Berners-Lee et. Al., January 2005. URL:http://www.ietf.org/rfc/rfc3986
[RFC 4122]	"A Universally Unique Identifier (UUID) URN Namespace", P. Leach, M. Mealling and R. Salz, July 2005. URL:http://www.ietf.org/rfc/rfc4122
[RFC 4566]	"SDP: Session Description Protocol", M. Handley, V. Jacobson and C. Perkins, July 2006. URL:http://www.ietf.org/rfc/rfc4566.txt

[RFC 4571] "Framing Real-time Transport Protocol (RTP) and RTP Control Protocol (RTCP) Packets over Connection-Oriented Transport", J. Lazzaro, July 2006.
URL:http://www.ietf.org/rfc/rfc4571.txt

[RFC 4585] "Extended RTP Profile for Real-time Transport Control Protocol (RTCP)-Based Feedback (RTP/AVPF)", J. Ott et al., July 2006.
URL:http://www.ietf.org/rfc/rfc4585.txt

[RFC 4702] "The Dynamic Host Configuration Protocol (DHCP) Client Fully Qualified Domain Name (FQDN) Option", M. Stapp, B. Volz and Y. Rekhter, October 2006.
URL:http://www.ietf.org/rfc/rfc4702

[RFC 4861] "Neighbor Discovery for IP version 6 (IPv6)", T. Narten et al., September 2007.
URL:http://www.ietf.org/rfc/rfc4861.txt

[RFC 4862] "IPv6 Stateless Address Auto configuration", S. Thomson, D. Narten and T. Jinmei, September 2007.
URL:http://www.ietf.org/rfc/rfc4862.txt

[SOAP 1.2, Part 1] "SOAP Version 1.2 Part 1: Messaging Framework", M. Gudgin (Ed) et. Al, April 2007.
URL:http://www.w3.org/TR/soap12-part1/

[SOAP 1.2, Part 2] "SOAP Version 1.2 Part 2: Adjuncts (Second Edition)", M. Gudgin (Ed) et. Al, April 2007.
URL:http://www.w3.org/TR/2007/REC-soap12-part2-20070427/

[WS-Addressing] "Web Services Addressing 1.0 – Core", M. Gudgin (Ed), M. Hadley (Ed) and T. Rogers (Ed), May 2006.
URL:http://www.w3.org/TR/ws-addr-core/#msgaddrprops

[WS-BaseNotification] "Web Services Base Notification 1.3", OASIS Standard, October 2006
URL:http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf

[WS-I BP 2.0] "Basic Profile Version 2.0 – Working Group Draft", C. Ferris (Ed), A. Karmarkar (Ed) and P. Yendluri (Ed), October 2007.
URL:http://www.ws-i.org/Profiles/BasicProfile-2_0(WGD).html

[WS-Discovery] "Web Services Dynamic Discovery (WS-Discovery)", J. Beatty et. Al., April 2005.
URL:http://specs.xmlsoap.org/ws/2005/04/discovery/ws-discovery.pdf

[WS-Security] "Web Services Security: SOAP Message Security 1.1 (WS-Security 2004)", OASIS Standard, February 2006.
URL:http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf

[WS-Topics] "Web Services Topics 1.3", OASIS Standard, 1 October 2006.
URL:http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-os.pdf

[WS-UsernameToken] "Web Services Security UsernameToken Profile 1.0", OASIS Standard, March 2004.
URL:http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-username-token-profile-1.0.pdf

[WSDL1.1] "Web Services Description Language (WSDL) 1.1", E. Christensen et. Al, March 2001.
URL:http://www.w3.org/TR/wsdl

[XML-Schema, Part 1] "XML Schema Part 1: Structures Second Edition", H. S. Thompson (Ed) et. Al, October 2004.
URL:http://www.w3.org/TR/xmlschema-1/

[XML-Schema, Part 2] "XML Schema Part 2: Datatypes Second Edition", P. V. Biron (ed) et. Al, October 2004.
URL:http://www.w3.org/TR/xmlschema-2/

2 Terms and Definitions

2.1 Conventions

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this specification are to be interpreted as described in [RFC2119].

These keywords are capitalized when used to unambiguously specify requirements over protocol and application features and behaviour that affect the interoperability and security of implementations. When these words are not capitalized, they are meant in their natural-language sense.

2.2 Definitions

Address	An address refers to a URI
Capability	The capability commands allow an NVC to ask for the services provided by an NVT.
Configuration Entity	A network video device media abstract component that is used to produce a media stream on the network, i.e. video and/or audio stream.
Media Profile	A media profile maps a video and/or audio source to a video and/or an audio encoder, PTZ and analytics configurations.
Network	A network is an interconnected group of devices communicating using the Internet protocol.
NVC Test Tool	Network Video Client Test tool that tests the Network Video Transmitter device compliance towards the ONVIF Core Specification v1.0
Proxy Server	A server that services the requests of its clients (NVC) by forwarding requests to other servers (NVT). A Proxy provides indirect network connections to its clients (NVC).
SOAP	SOAP is a lightweight protocol intended for exchanging structured information in a decentralized, distributed environment. It uses XML technologies to define an extensible messaging framework providing a message construct that can be exchanged over a variety of underlying protocols.
Switching Hub	A device for connecting multiple Ethernet devices together, making them act as a single network segment.
Target Service	An endpoint that makes itself available for discovery.
Tunneling	A proxy server that passes all requests and replies unmodified.

2.3 Abbreviations

AAC	Advanced Audio Coding
DUT	Device Under Test
DP	Discovery Proxy
DNS	Domain Name System
DHCP	Dynamic Host Configuration Protocol
HTTP	Hyper Text Transport Protocol
HTTPS	Hyper Text Transport Protocol over Secure Socket Layer
IP	Internet Protocol
IPv4	Internet Protocol version 4
IPv6	Internet Protocol version 6
JPEG	Joint Photographic Experts Group

MPEG-4	Moving Pictures Experts Group-4
NVT	Network Video Transmitter
NVC	Network Video Client
NTP	Network Time Protocol
POSIX	Portable Operating System Interface
PTZ	Pan/Tilt/Zoom
QVGA	Quarter Video Graphics Array
RTCP	RTP Control Protocol
RTSP	Real Time Streaming Protocol
RTP	Real-time Transport Protocol
SDP	Session Description Protocol
TCP	Transport Control Protocol
TTL	Time To Live
UTC	Coordinated Universal Time
USB	Universal Serial Bus
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 about 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 NVT Device

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.

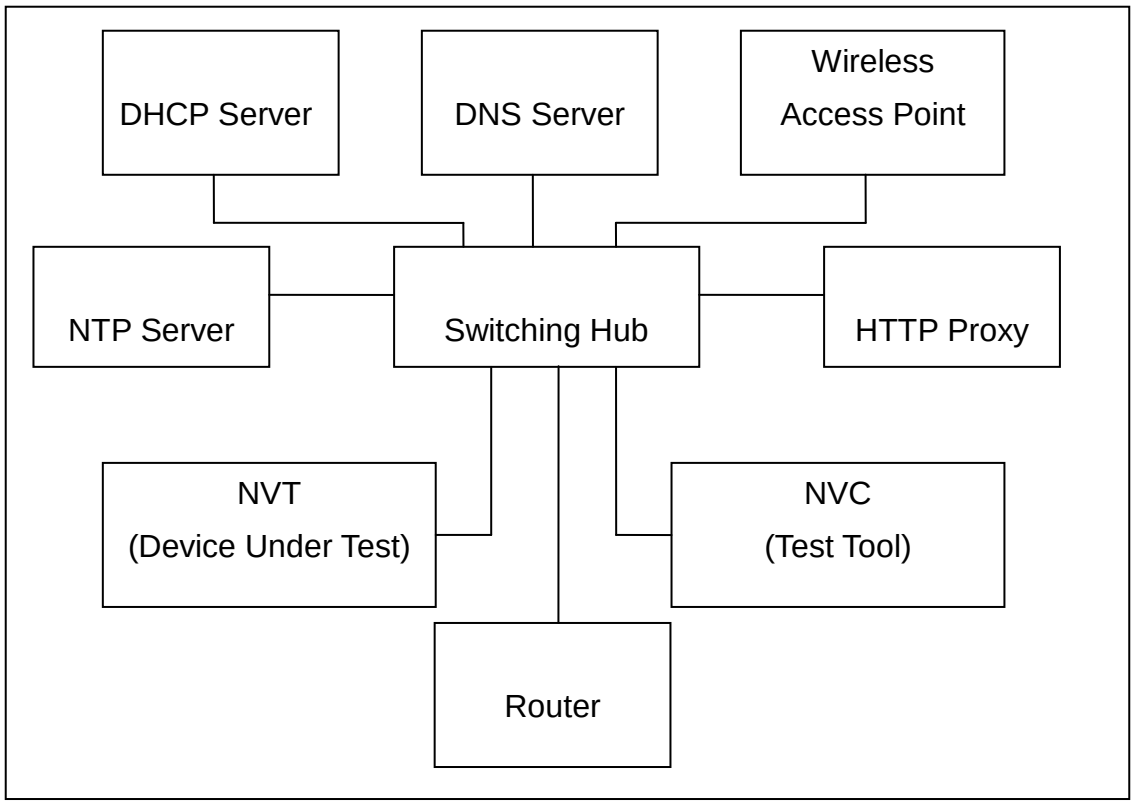


Figure 1: Test Configuration for NVT Device

NVT: Network Video Transmitter. Hereafter, this is referred to as DUT (Device Under Test).

NVC 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 tunneling over HTTP.

Wireless Access Point: provides wireless connectivity to the devices that support wireless connection.

DNS Server: provides DNS related information to the connected devices.

DHCP Server: provides IPv4 Address to the connected devices.

NTP Server: provides time synchronization between NVC 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 must be configured with an IPv4 address.
- The DUT must be IP reachable [in the test configuration].
- The DUT must be able to be discovered by the NVC Test Tool.
- The DUT must be configured with the time i.e. manual configuration of UTC time and if NTP is supported by DUT then NTP time must be synchronized with NTP Server.
- The DUT time and NVC Test tool time must be synchronized with each other either manually or by common NTP server.

3.3 Requirement Level

The general interpretation of the requirement levels is as defined in [RFC2119]. The following sections describe how the requirement levels affect the test procedure.

3.3.1 MUST

Test cases that cover parts of the [ONVIF Core] that are mandatory to implement in all ONVIF conformant products have the requirement level “MUST”. The test result for these test cases MUST be "PASSED" for the DUT to be ONVIF conformant.

3.3.2 MUST IF SUPPORTED

The requirement level “MUST IF SUPPORTED” is used for test cases that cover parts of the [ONVIF Core] that are mandatory to implement if and only if the DUT supports the referenced service, feature or functional block in any possible way.

If the DUT does support the referenced service, feature or functional block, then the test result MUST be "PASSED" for the DUT to be ONVIF conformant.

If the DUT does not support the referenced service, feature or functional block, then the DUT MUST correctly reply with a proper fault message to be ONVIF conformant. The test result in this case MUST be "DEVICE FEATURE NOT SUPPORTED BY NVT".

3.3.3 SHOULD, SHOULD IF SUPPORTED and OPTIONAL

The “SHOULD” level indicates that the service, functional block or feature, SHOULD be implemented by the DUT. The “SHOULD IF SUPPORTED” level indicates that the service, functional block or feature, SHOULD be implemented by the DUT if supported by the DUT in any way. The “OPTIONAL” level indicates that the service, functional block or feature, MAY or MAY NOT be implemented by the DUT. Failure to comply with these requirement levels is not a violation of the ONVIF Conformance

requirement. However, if the ONVIF support is implemented, then it MUST be done in conformance with the [ONVIF Core].

If the referenced part of the [ONVIF Core] has been implemented in the DUT, then the test result MUST be “PASSED” for the DUT to be ONVIF conformant.

If the referenced part of the [ONVIF Core] has not been implemented in the DUT, then the test should not be executed.

3.3.4 MUST IF IMPLEMENTED [A]

The requirement level “MUST IF IMPLEMENTED [A]” is used for test cases that cover parts of the [ONVIF Core] that are mandatory to implement if and only if the DUT implements ‘A’ in conformance with the [ONVIF Core].

Here ‘A’ refers to the service, feature or functional block in [ONVIF Core] which has requirement level of “SHOULD/ SHOULD IF SUPPORTED/ OPTIONAL” (Ex: IPv6).

If DUT implements ‘A’ in conformance with [ONVIF Core], then the test result MUST be “PASSED” for the DUT to be ONVIF conformant.

If DUT doesn’t implement ‘A’ in conformance with [ONVIF Core], then the test case should not be executed.

3.3.5 MUST IF SUPPORTED [A] & IMPLEMENTED [B]

The requirement level “MUST IF SUPPORTED [A] & IMPLEMENTED [B]” is used for test cases that cover parts of the [ONVIF Core] that are mandatory to implement if and only if the DUT supports ‘A’ in any possible way and the DUT implements ‘B’ in conformance with the [ONVIF Core].

Here ‘A’ refers to the service, feature or functional block in [ONVIF Core] that is mandatory to be implemented if DUT supports it (Ex: PTZ). ‘B’ refers to the service, feature or functional block in [ONVIF Core] that has requirement level of “SHOULD/ SHOULD IF SUPPORTED/ OPTIONAL” (Ex: IPv6).

If the DUT supports ‘A’ in any possible way and implements ‘B’ in conformance with [ONVIF Core], then the test result MUST be “PASSED” for the DUT to be ONVIF conformant.

If the DUT doesn’t support ‘A’ or doesn’t implement ‘B’ in conformance with [ONVIF Core], then the test case should not be executed.

3.3.6 SHOULD IF IMPLEMENTED [A]

The requirement level “SHOULD IF IMPLEMENTED [A]” is used for test cases that cover parts of the [ONVIF Core] that SHOULD be implemented if and only if the DUT implements ‘A’ in conformance with the [ONVIF Core].

Here ‘A’ refers to the service, feature or functional block in [ONVIF Core] which has requirement level of “SHOULD/ SHOULD IF SUPPORTED/ OPTIONAL” (Ex: IPv6).

Failure to comply with this requirement level is not a violation of the ONVIF Conformance requirement. However, if the ONVIF support is implemented, then it MUST be done in conformance with the [ONVIF Core].

If DUT implements ‘A’ in conformance with [ONVIF Core] and the reference part of [ONVIF Core] is implemented, then the test result MUST be “PASSED” for the DUT to be ONVIF conformant.

If DUT doesn’t implement ‘A’ in conformance with [ONVIF Core] or the reference part of [ONVIF Core] is not implemented, then the test case should not be executed.

3.3.7 SHOULD IF SUPPORTED [A] & IMPLEMENTED [B]

The requirement level “SHOULD IF SUPPORTED [A] & IMPLEMENTED [B]” is used for test cases that cover parts of the [ONVIF Core] that SHOULD be implemented if and only if the DUT supports ‘A’ in any possible way and the DUT implements ‘B’ in conformance with the [ONVIF Core].

Here ‘A’ refers to the service, feature or functional block in [ONVIF Core] that is mandatory to be implemented if DUT supports it (Ex: PTZ). ‘B’ refers to the service, feature or functional block in [ONVIF Core] that has requirement level of “SHOULD/ SHOULD IF SUPPORTED/ OPTIONAL” (Ex: IPv6).

Failure to comply with this requirement level is not a violation of the ONVIF Conformance requirement. However, if the ONVIF support is implemented, then it MUST be done in conformance with the [ONVIF Core].

If the DUT supports ‘A’ in any possible way, implements ‘B’ in conformance with [ONVIF Core] and implements the referenced part of [ONVIF Core], then the test result MUST be “PASSED” for the DUT to be ONVIF conformant.

If the DUT doesn’t support ‘A’ or doesn’t implement ‘B’ in conformance with [ONVIF Core] or doesn’t implement the referenced part of [ONVIF Core], then the test case should not be executed.

3.4 Test Policy

This section describes the test policies specific to the test case execution of each functional block.

The DUT (NVT) must adhere to the test policies defined in this section.

3.4.1 IP Configuration

- The device under test must be discovered by the NVC device that exists in the testing environment.
- The device under test must support SetNetworkInterfaces method.
- The device under test that supports Link-Local address of IPv4 must support SetZeroConfiguration method.
- The following tests are performed about IPv4.
  - Static IP configuration
  - Dynamic IP configuration of Link-Local address
  - Dynamic IP configuration (DHCP)
- The following tests are performed about IPv6.
  - Stateless IP configuration which accepts Router Advertisement.
  - Stateless IP configuration which uses Neighbour Discovery.
  - Stateful IP configuration (DHCPv6)
- The device under test must have at least one network interface that gives IPv4 connectivity. And it should have at least one network interface that gives IPv6 connectivity.
- The device under test that has multiple network interfaces (Wired Ethernet i.e. 802.3af and Wireless Ethernet i.e. 802.11a/b/g/n), initial testing will be performed on the Wired

Ethernet network interface. After completion of all testing on the Wired Ethernet network interface, all tests shall be repeated on Wireless Ethernet network interface.

- ONVIF Test Specification restricts all testing to Wired Ethernet and/or Wireless Ethernet network interface, other interfaces like USB, Bluetooth etc are outside the scope of the testing.

Please refer to Section 4 for IP Configuration Test Cases.

3.4.2 Device Discovery

- The device under test must be discovered by the NVC device that exists in the testing environment.
- Failure to discover the device on the network constitutes failure of the test procedure.
- Failure to locate the device services on the network constitutes failure of the test procedure.
- Failure to select the device for interaction constitutes failure of the test procedure.
- Failure to exist of the namespace of defined Core Specification constitutes failure of the test procedure.
- In certain test cases, the NVC may check the discovery mode and change it to “discoverable” to do the test. At the end of the test procedure it resets the discovery mode value.

Please refer to Section 5 for Device Discovery Test Cases.

3.4.3 Device Management

- The device under test must demonstrate device, media and event capability. A NVT that does not display mandatory device capability constitutes failure of test procedure.
- Some commands like CreateUsers, SetUser, etc may have restricted access. In that case NVC should execute the test cases in the administrative mode.
- If DUT does not support PTZ, then (GET CAPABILITIES for PTZ) MUST be responded with SOAP 1.2 fault message (env:Receiver, ter:ActionNotSupported, ter:NoSuchService).

Please refer to Annex A.2 for valid host name.

Please refer to Section 6 for Device Management Test cases.

3.4.4 Media Configuration

- Prior to the execution of Media Configuration test cases, DUT must be discovered by NVC and it must demonstrate media capabilities to NVC using device management service.
- DUT must support at least one media profile with Video Configuration. Video Configuration must include video source and video encoder media entities.
- DUT must support JPEG QVGA encoding.
- NVC user must explicitly specify the optional media formats supported by DUT.

- NVC user must explicitly specify if the DUT supports Audio and PTZ.
- In certain test cases, NVC may create new media configuration (i.e. media profile and media entities). In such cases, the test procedure will delete those modified configuration at the end of the test procedure.
- DUT must allow for creation of at least one media profile by NVC. In certain test cases, NVC may create new media configuration (i.e. media profile and media entities). In such cases, the test procedure will delete those modified configuration at the end of the test procedure.
- DUT should respond with proper response message for all SOAP actions. Sending fault messages such as “ter:ConfigurationConflict” will be treated as FAILURE of the test cases.

Please refer to Section 7 for Media Configuration Test Cases.

3.4.5 Real Time Streaming

- Real time streaming test case execution would need the successful execution of some of the Media Configuration test cases. So, Media Configuration features must be implemented successfully in order to execute the Real Time Streaming test cases.
- NVC user must explicitly specify the optional transport protocols supported by DUT.
- NVC and DUT time should be synchronized for media streaming.
- Real time streaming testing will test only one media stream at a time.
- Poor streaming test is out side the scope of the ONVIF Test Specification.

Please refer to Annex A.10 for the correct interpretation of StreamSetup syntax.

Please refer to Section 8 for Real Time Streaming Test Cases.

3.4.6 Event Handling

- Prior to the execution of Event handling test cases, DUT MUST be discovered by NVC and it MUST demonstrate event capabilities to NVC using the device management service.
- If the DUT supports “property events” NVC uses the SetSynchronizationPoint method from the event service to trigger events for testing Basic Notification Interface, Realtime Pullpoint Interface; NVC uses the SetSynchronizationPoint from the Media service to trigger events for testing metadata streaming.
- If the DUT does not support “property events”, the event should be triggered manually.
- NVC and DUT time SHOULD be synchronized.
- In certain test cases the Test Tool MAY create a Subscription Manager representing the subscription. In such cases the procedure will take care that all new created Subscription Managers are deleted at the end of the test procedure.
- In certain test cases the Test Tool MAY create or change media entities (e.g. add a MetadataConfiguration to a profile).In such cases the procedure will delete those modifications at the end of the test procedure.
- The Test Tool will only test one Notification Stream at a time; poor streaming performance is out of scope.

Please refer to Section 9 for Event handling Test Cases.

3.4.7 PTZ Control

- To start with NVC shall check device capabilities of PTZ. If the DUT doesn't have PTZ capability, this test case will skip.
- The device under test must support at-least one media profile with PTZ configuration. Moreover, the DUT must include video source configuration and video encoder configuration in the same media profile to see the video and to confirm movement. A PTZ configuration must include a PTZ node.
- Poor PTZ performance test is outside the scope of the ONVIF Test Specification.
- In certain test cases, NVC may register new preset position into PTZ configuration. In such cases, the test procedure will delete those modified configuration at the end of the test procedure.
- If DUT does not support PTZ Configuration commands (ex. GetConfigurations, AbsoluteMove) then it MUST respond to the request with SOAP 1.2 fault message (ActionNotSupported).

Please refer to Section 10 for PTZ Configuration Test Cases.

3.4.8 Security

- The DUT MUST support WS-Security User token profile. Consequently the DUT MUST support user profiles that conform to the User token profile and handling of these users via Device Management.
- The details of Access rights and Access policies are outside the scope of this document. However, an NVC MUST be able to access any given part of any given service supported by the NVT with a user with Administrator rights.

Please refer to Section 11 for Security Test Cases.

4 IP Configuration Test Cases

4.1 IPv4

4.1.1 NVT IPV4 STATIC IP

Test Label: IP Configuration NVT IPv4 Static IP Configuration

ONVIF Core Specification Coverage: 6. IP Configuration

Device Type: NVT

Command Under Test: None

WSDL Reference: devicemgmt.wsdl

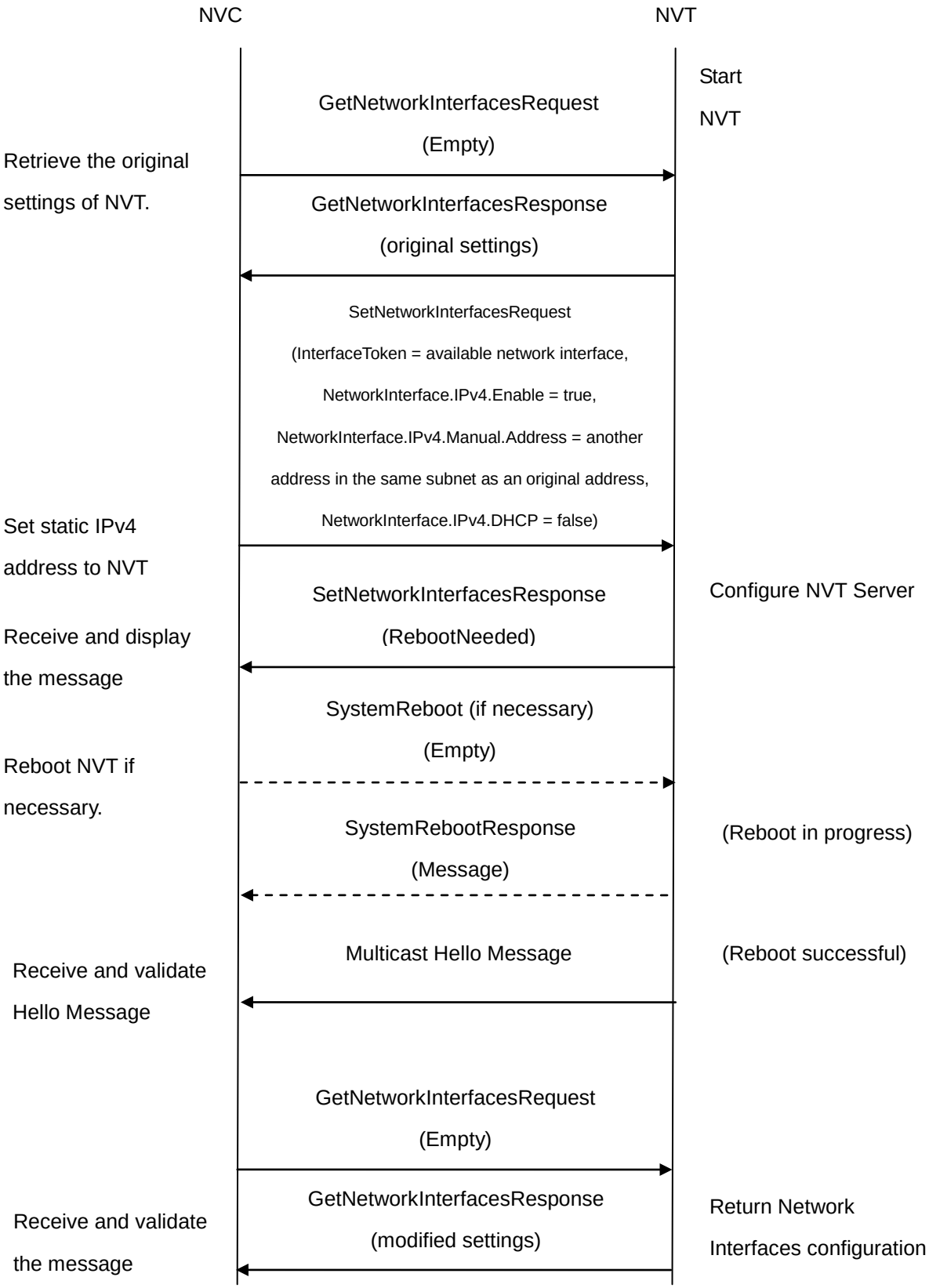
Requirement Level: MUST

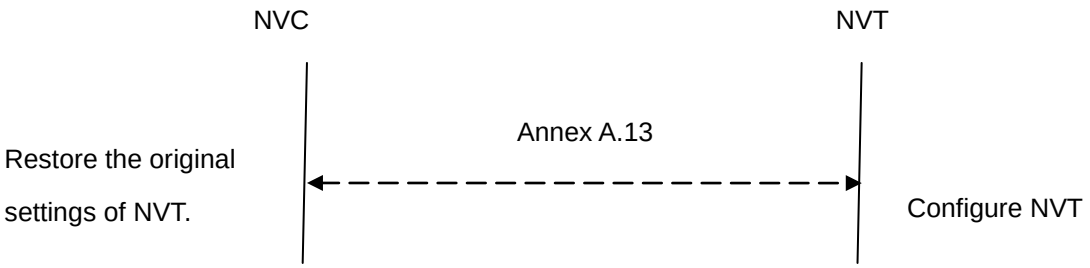
Test Purpose: To test IPv4 Static IP Configuration.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
GetNetworkInterfacesRequest
(Empty)
Retrieve the original settings of NVT.
GetNetworkInterfacesResponse
(original settings)
SetNetworkInterfacesRequest
(InterfaceToken = available network interface,
NetworkInterface.IPv4.Enable = true,
NetworkInterface.IPv4.Manual.Address = another address in the same subnet as an original address,
NetworkInterface.IPv4.DHCP = false)
Set static IPv4 address to NVT
SetNetworkInterfacesResponse
(RebootNeeded)
Configure NVT Server
Receive and display the message
SystemReboot (if necessary)
(Empty)
Reboot NVT if necessary.
SystemRebootResponse
(Message)
(Reboot in progress)
Multicast Hello Message
(Reboot successful)
Receive and validate Hello Message
GetNetworkInterfacesRequest
(Empty)
GetNetworkInterfacesResponse
(modified settings)
Return Network Interfaces configuration
Receive and validate the message



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNetworkInterfacesRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetNetworkInterfacesRequest message to set static IPv4 address to NVT (InterfaceToken = available network interface, NetworkInterface.IPv4.Enabled = true, NetworkInterface.IPv4.Manual.Address = another address in the same subnet as an original address, NetworkInterface.IPv4.DHCP = false).
5. NVT will return SetNetworkInterfacesResponse message.
6. If necessary, NVC will invoke SystemReboot message to restart NVT. Otherwise, continue to step-8.
7. NVT will return SystemRebootResponse message.
8. NVT will send Multicast Hello message from newly configured address.
9. NVC will receive and validate Hello message sent from newly configured address by NVT.
10. NVC will invoke GetNetworkInterfacesRequest message to newly configured address to retrieve the modified settings of NVT.
11. NVC will receive and validate GetNetworkInterfacesResponse message sent from newly configured address by NVT.
12. NVC will restore the original settings by following the procedure mentioned in Annex A.13.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNetworkInterfacesResponse message.

The DUT did not send SystemRebootResponse message.

The DUT did not send Hello message after IP configuration change.

The DUT did not send GetNetworkInterfacesResponse message.

The DUT did not send correct network interface information (i.e. NetworkInterface[token = available network interface token, IPv4.Enabled = true, IPv4.Config.Manual = newly configured address ,IPv4.DHCP = false]).in GetNetworkInterfacesResponse message.

4.1.2 NVT IPV4 LINK LOCAL ADDRESS

Test Label: IP Configuration NVT IPv4 Link-Local Address Configuration

ONVIF Core Specification Coverage: 6. IP Configuration, 8.2.16 Set zero configuration

Device Type: NVT

Command Under Test: SetZeroConfiguration, GetZeroConfiguration

WSDL Reference: devicemgmt.wsdl

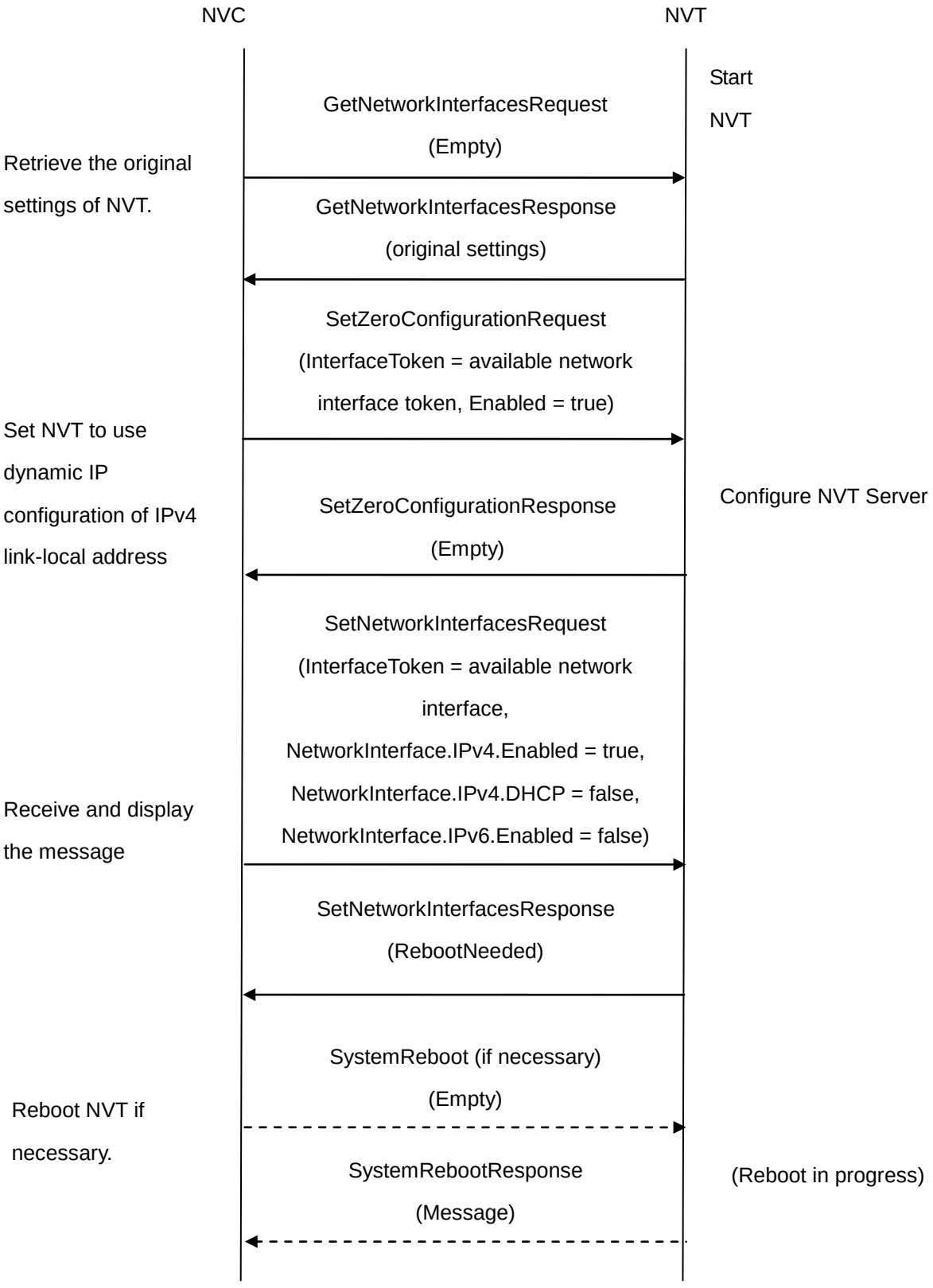
Requirement Level: SHOULD

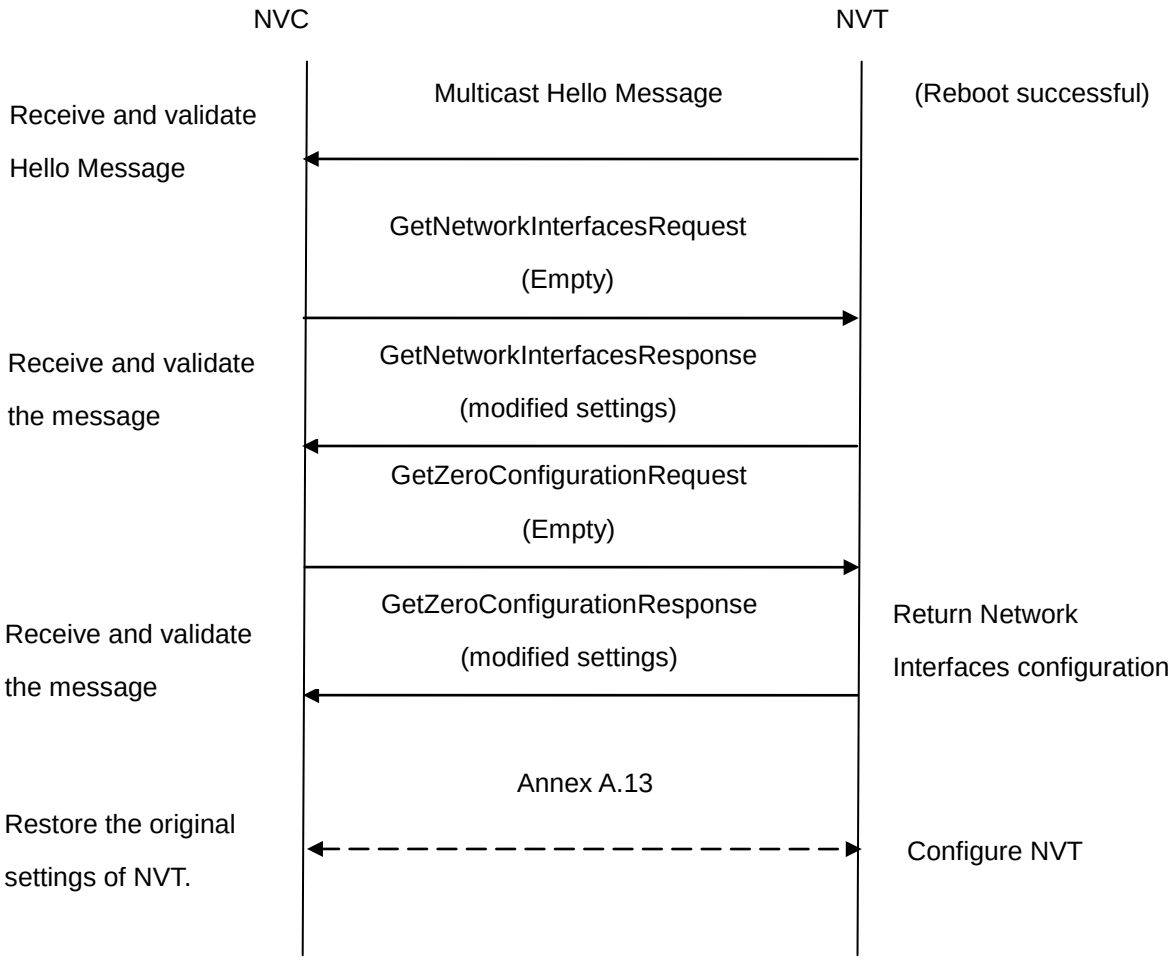
Test Purpose: To test IPv4 Link-Local Address Configuration.

Pre-Requisite: Dynamic IP configuration as per RFC 3927 is supported by NVT.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
GetNetworkInterfacesRequest
(Empty)
Retrieve the original settings of NVT.
GetNetworkInterfacesResponse
(original settings)
SetZeroConfigurationRequest
(InterfaceToken = available network interface token, Enabled = true)
Set NVT to use dynamic IP configuration of IPv4 link-local address
SetZeroConfigurationResponse
(Empty)
Configure NVT Server
SetNetworkInterfacesRequest
(InterfaceToken = available network interface,
NetworkInterface.IPv4.Enabled = true,
NetworkInterface.IPv4.DHCP = false,
NetworkInterface.IPv6.Enabled = false)
Receive and display the message
SetNetworkInterfacesResponse
(RebootNeeded)
SystemReboot (if necessary)
(Empty)
Reboot NVT if necessary.
SystemRebootResponse
(Message)
(Reboot in progress)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNetworkInterfacesRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetZeroConfigurationRequest message to set NVT to use dynamic IP configuration of IPv4 link-local address (InterfaceToken = available network interface token, Enabled = true).
5. NVT will return SetZeroConfigurationResponse message.
6. NVC will invoke SetNetworkInterfacesRequest message to set static IPv4 address to NVT (InterfaceToken = available network interface, NetworkInterface.IPv4.Enabled = true, NetworkInterface.IPv4.DHCP = false, NetworkInterface.IPv6.Enabled = false).
7. NVT will return SetNetworkInterfacesResponse message.

8. If necessary, NVC will invoke SystemReboot message to restart NVT. Otherwise, continue to step-10.
9. NVT will return SystemRebootResponse message.
10. NVT will send Multicast Hello message from newly configured address.
11. NVC will receive and validate Hello message sent from newly configured address by NVT.
12. NVC will invoke GetNetworkInterfacesRequest message to newly configured address to retrieve the modified settings of NVT.
13. NVC will receive and validate GetNetworkInterfacesResponse message sent from newly configured address by NVT.
14. NVC will invoke GetZeroConfigurationRequest message to newly configured address to retrieve the modified settings of NVT.
15. NVC will receive and validate GetZeroConfigurationResponse message sent from newly configured address by NVT.
16. NVC will restore the original settings by following the procedure mentioned in Annex A.13.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNetworkInterfacesResponse message.

The DUT did not send SystemRebootResponse message.

The DUT did not send Hello message after IP configuration change

The DUT did not send SetZeroConfigurationResponse message

The DUT did not send GetNetworkInterfacesResponse message.

The DUT did not send GetZeroConfigurationResponse message.

The DUT did not send correct network interface information (i.e. NetworkInterface[token = available network interface token, IPv4.Enabled = true, IPv4.Config.LinkLocal = new IP address]).in GetNetworkInterfacesResponse message.

The DUT did not send correct zero configuration information (i.e. InterfaceToken = available network interface token, Enabled = true, Address = new IP address).in GetZeroConfigurationResponse message.

4.1.3 NVT IPV4 DHCP

Test Label: IP Configuration NVT IPv4 DHCP Configuration

ONVIF Core Specification Coverage: 6. IP Configuration

Device Type: NVT

Command Under Test: None

WSDL Reference: devicemgmt.wsdl

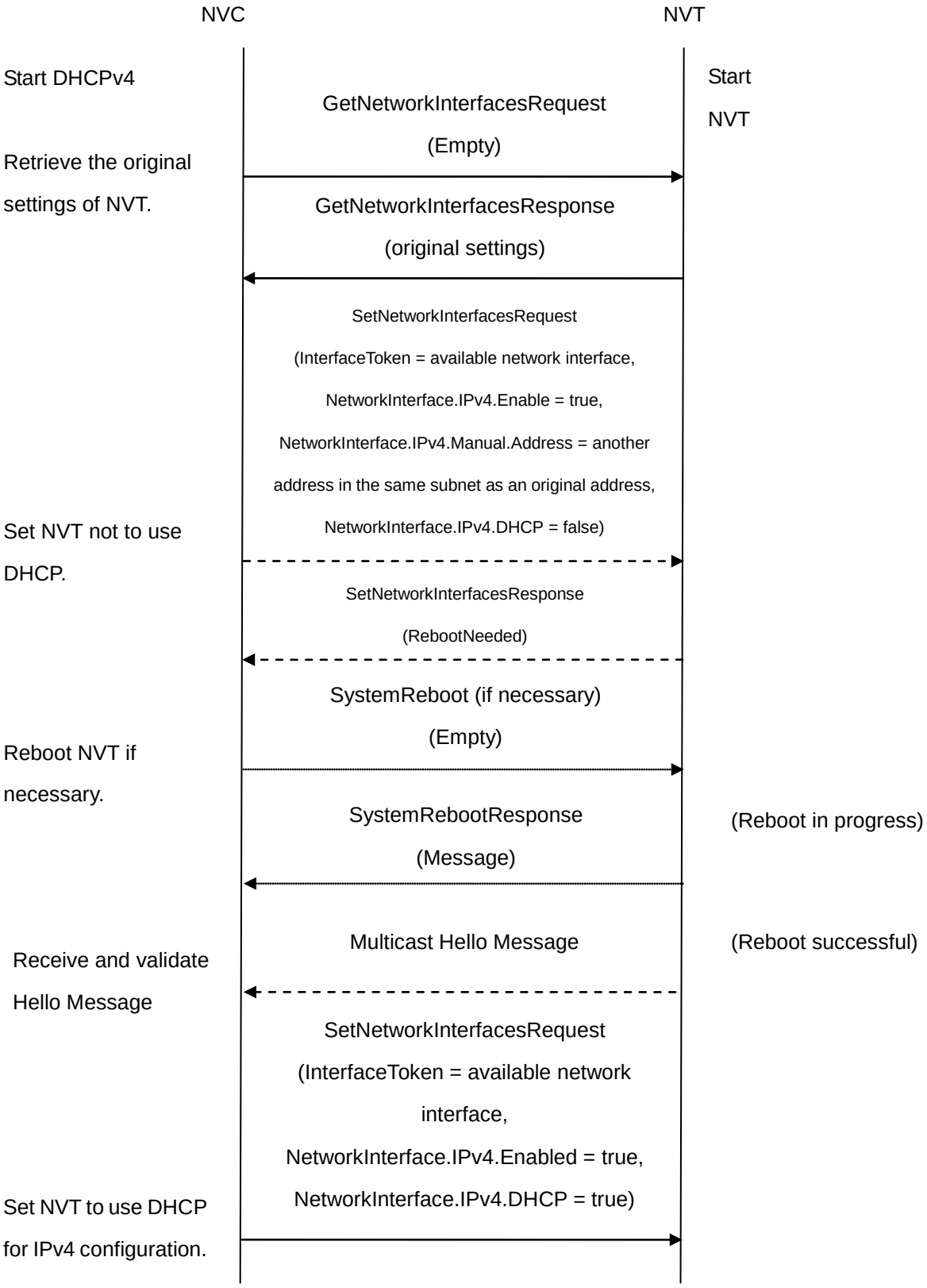
Requirement Level: MUST

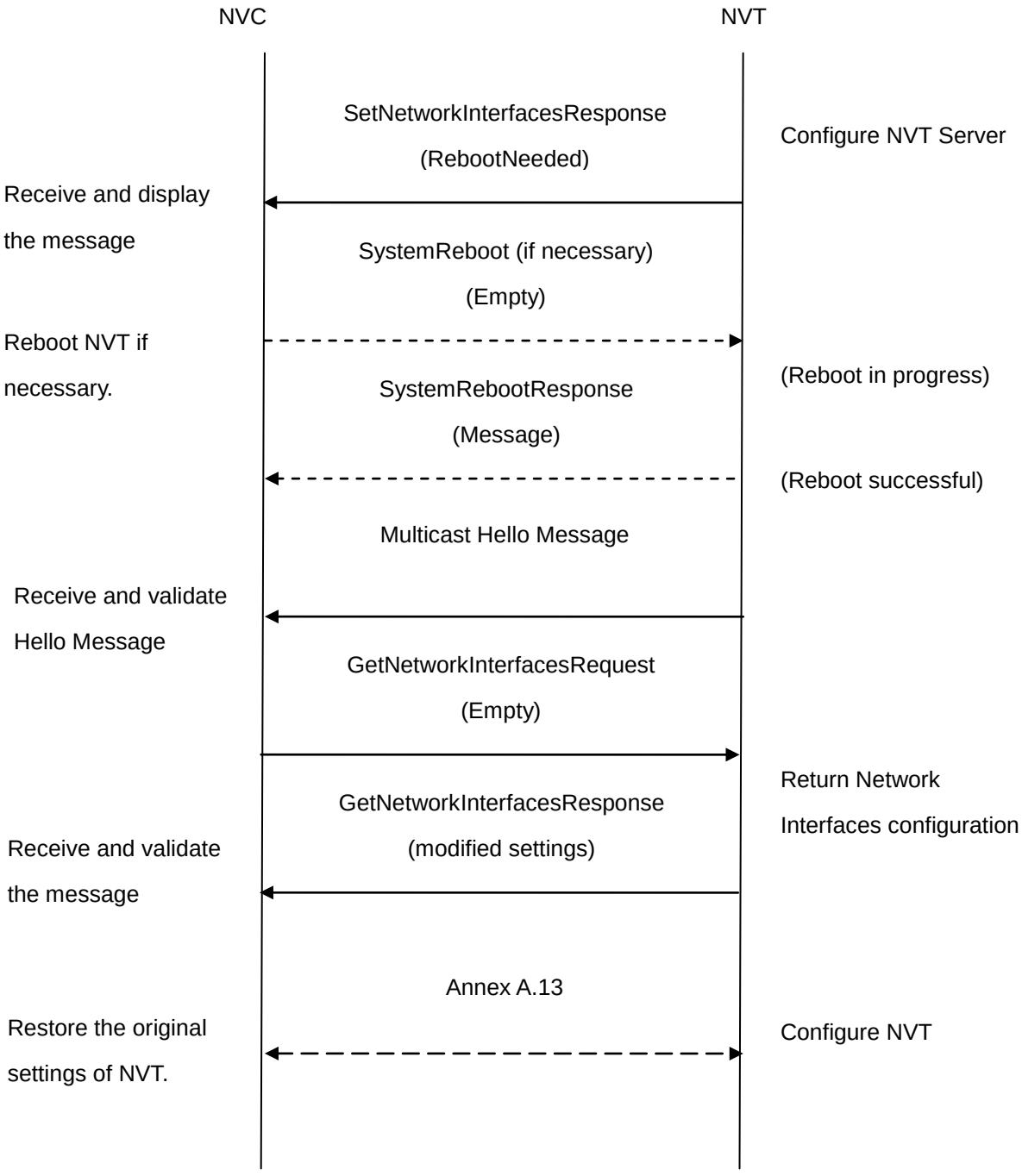
Test Purpose: To test IPv4 DHCP Configuration.

Pre-Requisite: None.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start DHCPv4
Start NVT
GetNetworkInterfacesRequest
(Empty)
Retrieve the original settings of NVT.
GetNetworkInterfacesResponse
(original settings)
SetNetworkInterfacesRequest
(InterfaceToken = available network interface,
NetworkInterface.IPv4.Enable = true,
NetworkInterface.IPv4.Manual.Address = another address in the same subnet as an original address,
NetworkInterface.IPv4.DHCP = false)
Set NVT not to use DHCP.
SetNetworkInterfacesResponse
(RebootNeeded)
SystemReboot (if necessary)
(Empty)
Reboot NVT if necessary.
SystemRebootResponse
(Message)
(Reboot in progress)
Multicast Hello Message
(Reboot successful)
Receive and validate Hello Message
SetNetworkInterfacesRequest
(InterfaceToken = available network interface,
NetworkInterface.IPv4.Enabled = true,
NetworkInterface.IPv4.DHCP = true)
Set NVT to use DHCP for IPv4 configuration.



Test Procedure:

1. Start DHCPv4 server
2. Start an NVC.
3. Start an NVT.

4. NVC will invoke GetNetworkInterfacesRequest message to retrieve the original settings of NVT.
5. If NetworkInterface.IPv4.DHCP == true in the original settings, NVC will invoke SetNetworkInterfacesRequest message to set NVT not to use DHCP (InterfaceToken = available network interface, NetworkInterface.IPv4.Enabled = true, NetworkInterface.IPv4.Manual.Address = another address in the same subnet as an original address, NetworkInterface.IPv4.DHCP = false, NetworkInterface.IPv6.Enable = false). Otherwise, continue to step-8.
6. If necessary, NVC will invoke SystemReboot message to restart NVT.
7. NVT will send Multicast Hello message from newly configured address.
8. NVC will invoke SetNetworkInterfacesRequest message to set NVT to use DHCP (InterfaceToken = available network interface, NetworkInterface.IPv4.Enabled = true, NetworkInterface.IPv4.Manual = empty, NetworkInterface.IPv4.DHCP = true, NetworkInterface.IPv6.Enable = false).
9. NVT will return SetNetworkInterfacesResponse message.
10. If necessary, NVC will invoke SystemReboot message to restart NVT. Otherwise, continue to step-12.
11. NVT will return SystemRebootResponse message.
12. NVT will send Multicast Hello message from newly configured address.
13. NVC will receive and validate Hello message sent from newly configured address by NVT.
14. NVC will invoke GetNetworkInterfacesRequest message to newly configured address to retrieve the modified settings of NVT.
15. NVC will receive and validate GetNetworkInterfacesResponse message sent from newly configured address by NVT.
16. If the modified settings are different from original settings then NVC will restore the original settings by following the procedure mentioned in Annex A.13.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNetworkInterfacesResponse message.

The DUT did not send SystemRebootResponse message.

The DUT did not send Hello message after IP configuration change.

The DUT did not send GetNetworkInterfacesResponse message.

The DUT did not send correct network interface information (i.e. NetworkInterface[token = available network interface token, IPv4.Enabled = true, IPv4.Config.FromDHCP = new IP address]).in GetNetworkInterfacesResponse message.

4.2 IPv6

4.2.1 NVT IPV6 STATIC IP

Test Label: IP Configuration NVT IPv6 Static IP Configuration

ONVIF Core Specification Coverage: 6. IP Configuration

Device Type: NVT

Command Under Test: None

WSDL Reference: devicemgmt.wsdl

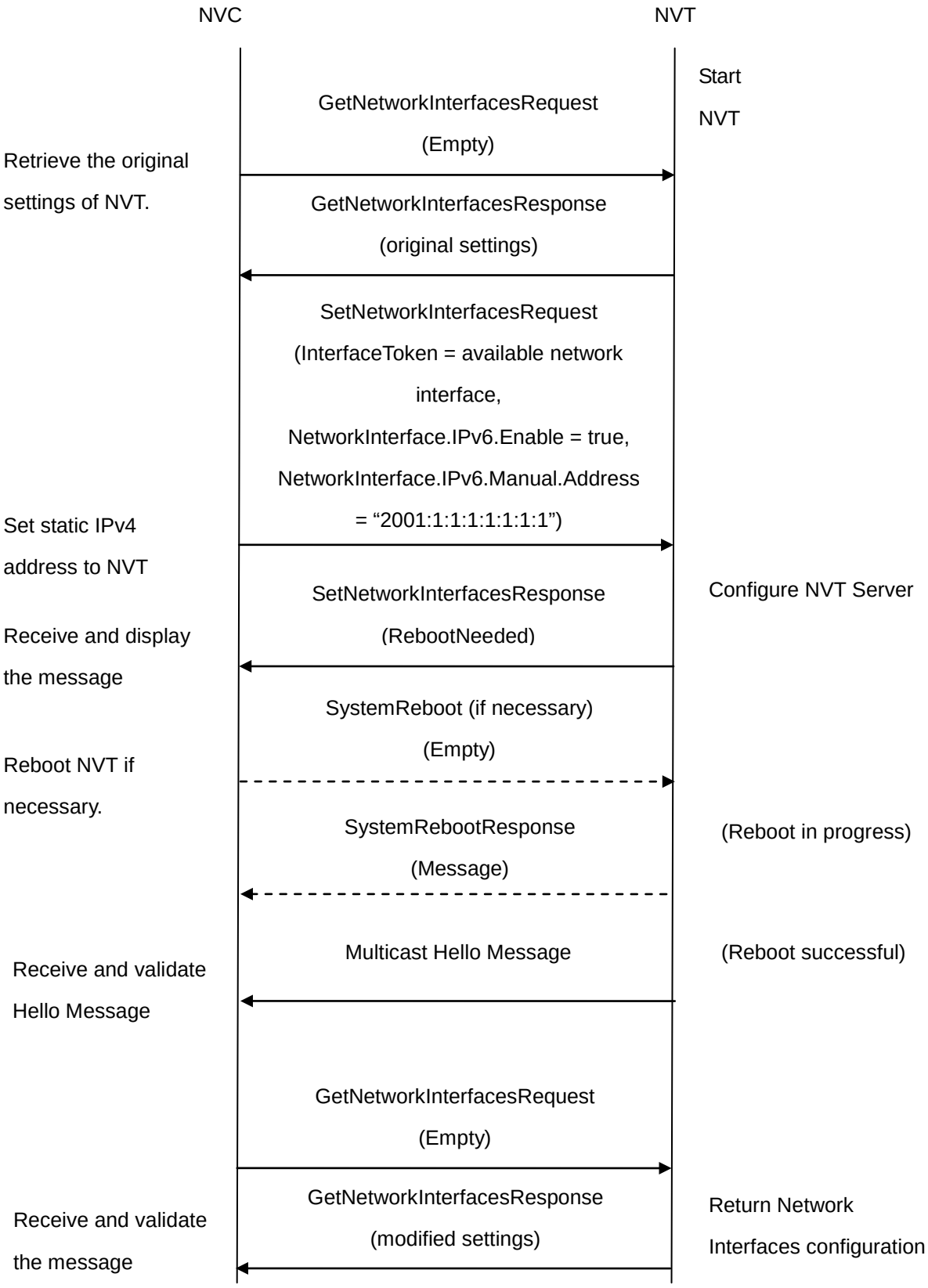
Requirement Level: MUST IF IMPLEMENTED (IPv6)

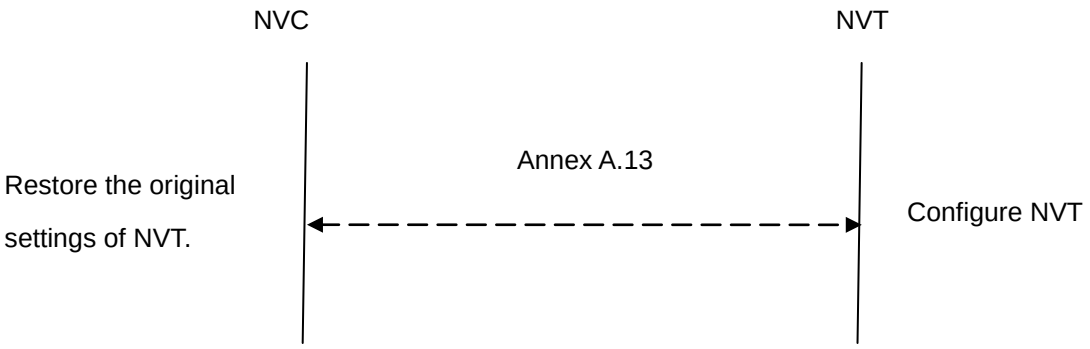
Test Purpose: To test IPv6 Static IP Configuration.

Pre-Requisite: IPv6 is implemented by NVT.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
GetNetworkInterfacesRequest
(Empty)
Retrieve the original settings of NVT.
GetNetworkInterfacesResponse
(original settings)
SetNetworkInterfacesRequest
(InterfaceToken = available network interface,
NetworkInterface.IPv6.Enable = true,
NetworkInterface.IPv6.Manual.Address = “2001:1:1:1:1:1:1:1”)
Set static IPv4 address to NVT
SetNetworkInterfacesResponse
(RebootNeeded)
Configure NVT Server
Receive and display the message
SystemReboot (if necessary)
(Empty)
Reboot NVT if necessary.
SystemRebootResponse
(Message)
(Reboot in progress)
Multicast Hello Message
(Reboot successful)
Receive and validate Hello Message
GetNetworkInterfacesRequest
(Empty)
GetNetworkInterfacesResponse
(modified settings)
Return Network Interfaces configuration
Receive and validate the message



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNetworkInterfacesRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetNetworkInterfacesRequest message to set static IPv6 address to NVT (InterfaceToken = available network interface, NetworkInterface.IPv6.Enabled = true, NetworkInterface.IPv6.Manual.Address = “2001:1:1:1:1:1:1:1”).
5. NVT will return SetNetworkInterfacesResponse message.
6. If necessary, NVC will invoke SystemReboot message to restart NVT. Otherwise, continue to step-8.
7. NVT will return SystemRebootResponse message.
8. NVT will send Multicast Hello message from newly configured address.
9. NVC will receive and validate Hello message sent from newly configured address by NVT.
10. NVC will invoke GetNetworkInterfacesRequest message to newly configured address to retrieve the modified settings of NVT.
11. NVC will receive and validate GetNetworkInterfacesResponse message sent from newly configured address by NVT.
12. NVC will restore the original settings by following the procedure mentioned in Annex A.13.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNetworkInterfacesResponse message.

The DUT did not send SystemRebootResponse message.

The DUT did not send Hello message after IP configuration change.

The DUT did not send GetNetworkInterfacesResponse message.

The DUT did not send correct network interface information (i.e. NetworkInterface[token = available network interface token, IPv6.Enabled = true, IPv6.Config.Manual = “2001:1:1:1:1:1:1:1”]).in GetNetworkInterfacesResponse message.

4.2.2 NVT IPV6 STATELESS IP CONFIGURATION - ROUTER ADVERTISEMENT

Test Label: IP Configuration NVT IPv6 Stateless IP Configuration Which Accepts Router Advertisement.

ONVIF Core Specification Coverage: 6. IP Configuration

Device Type: NVT

Command Under Test: None

WSDL Reference: devicemgmt.wsdl

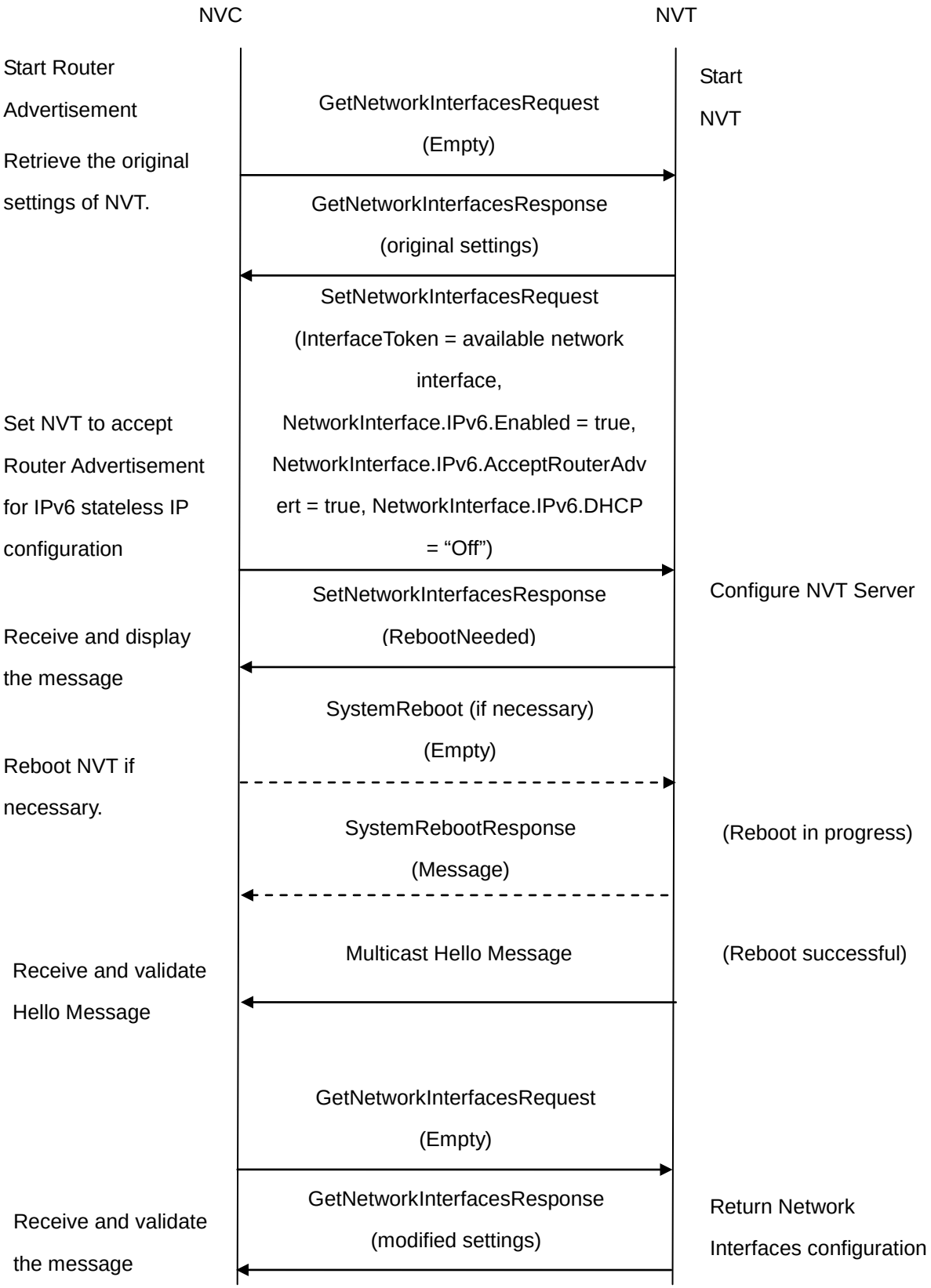
Requirement Level: MUST IF IMPLEMENTED (IPv6)

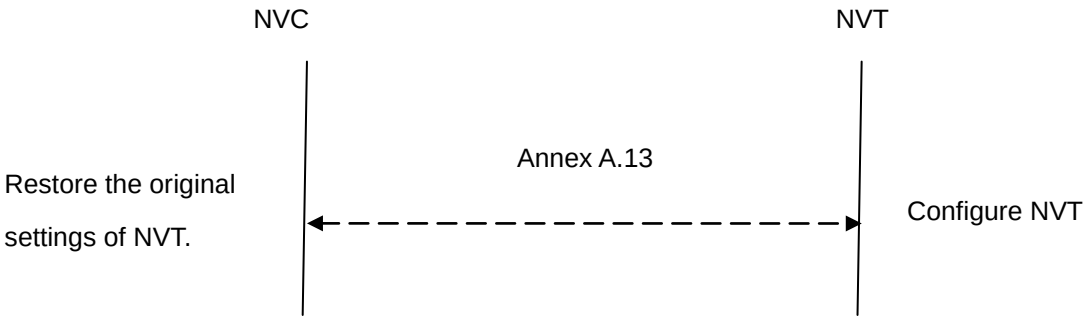
Test Purpose: To test IPv6 Stateless IP Configuration which Accepts Router Advertisement.

Pre-Requisite: IPv6 is implemented by NVT.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start Router Advertisement
Retrieve the original settings of NVT.
GetNetworkInterfacesRequest
(Empty)
Start NVT
GetNetworkInterfacesResponse
(original settings)
SetNetworkInterfacesRequest
(InterfaceToken = available network interface,
NetworkInterface.IPv6.Enabled = true,
NetworkInterface.IPv6.AcceptRouterAdvert = true, NetworkInterface.IPv6.DHCP = “Off”)
Set NVT to accept Router Advertisement for IPv6 stateless IP configuration
SetNetworkInterfacesResponse
(RebootNeeded)
Configure NVT Server
Receive and display the message
SystemReboot (if necessary)
(Empty)
Reboot NVT if necessary.
SystemRebootResponse
(Message)
(Reboot in progress)
Multicast Hello Message
(Reboot successful)
Receive and validate Hello Message
GetNetworkInterfacesRequest
(Empty)
GetNetworkInterfacesResponse
(modified settings)
Return Network Interfaces configuration
Receive and validate the message



Test Procedure:

1. Start Router Advertisement
2. Start an NVC.
3. Start an NVT.
4. NVC will invoke GetNetworkInterfacesRequest message to retrieve the original settings of NVT.
5. NVC will invoke SetNetworkInterfacesRequest message to set NVT to accept Router Advertisement for IPv6 stateless IP configuration (InterfaceToken = available network interface, NetworkInterface.IPv6.Enabled = true, NetworkInterface.IPv6.AcceptRouterAdvert = true, NetworkInterface.IPv6.DHCP = "Off").
6. NVT will return SetNetworkInterfacesResponse message.
7. If necessary, NVC will invoke SystemReboot message to restart NVT. Otherwise, continue to step-9.
8. NVT will return SystemRebootResponse message.
9. NVT will send Multicast Hello message from newly configured address.
10. NVC will receive and validate Hello message sent from newly configured address by NVT.
11. NVC will invoke GetNetworkInterfacesRequest message to newly configured address to retrieve the modified settings of NVT.
12. NVC will receive and validate GetNetworkInterfacesResponse message sent from newly configured address by NVT.
13. NVC will restore the original settings by following the procedure mentioned in Annex A.13.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNetworkInterfacesResponse message.

The DUT did not send SystemRebootResponse message.

The DUT did not send Hello message after IP configuration change.

The DUT did not send GetNetworkInterfacesResponse message.

The DUT did not send correct network interface information (i.e. NetworkInterface[token = available network interface token, IPv6.Enabled = true, IPv6.Config.FromRA = new IP address]).in GetNetworkInterfacesResponse message.

4.2.3 NVT IPV6 STATELESS IP CONFIGURATION - NEIGHBOUR DISCOVERY

Test Label: IP Configuration NVT IPv6 Stateless IP Configuration Which Uses Neighbour Discovery.

ONVIF Core Specification Coverage: 6. IP Configuration

Device Type: NVT

Command Under Test: None

WSDL Reference: devicemgmt.wsdl

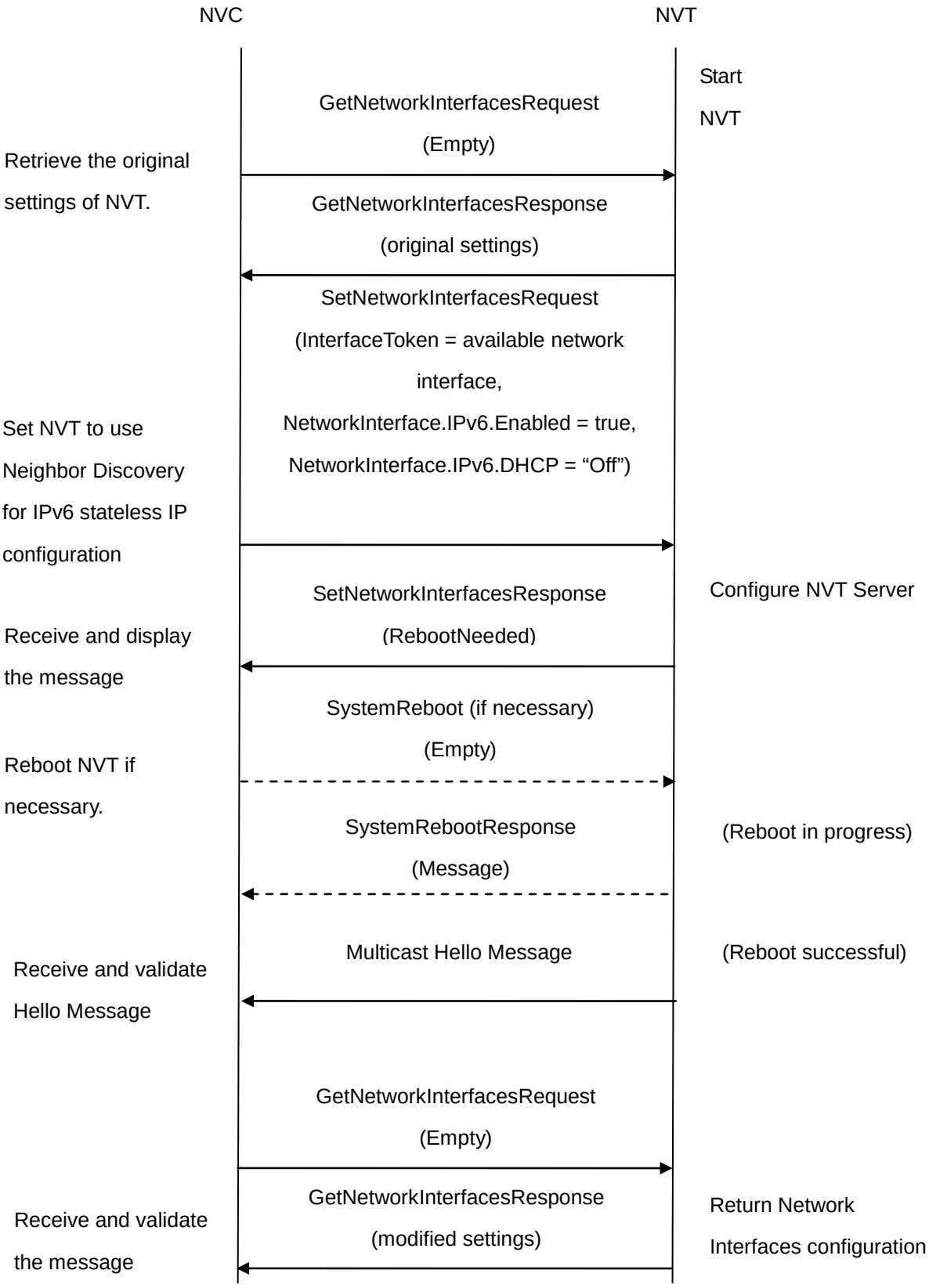
Requirement Level: MUST IF IMPLEMENTED (IPv6)

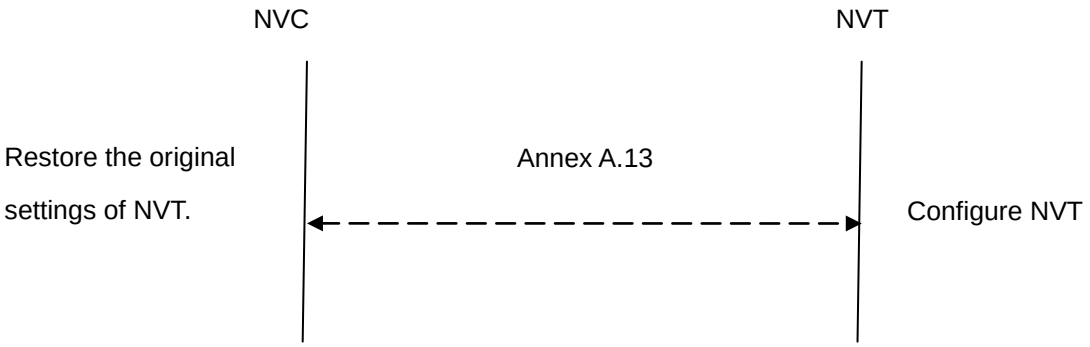
Test Purpose: To test IPv6 Stateless IP Configuration which use Neighbour Discovery

Pre-Requisite: IPv6 is implemented by NVT.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
GetNetworkInterfacesRequest
(Empty)
Retrieve the original settings of NVT.
GetNetworkInterfacesResponse
(original settings)
SetNetworkInterfacesRequest
(InterfaceToken = available network interface,
NetworkInterface.IPv6.Enabled = true,
NetworkInterface.IPv6.DHCP = “Off”)
Set NVT to use Neighbor Discovery for IPv6 stateless IP configuration
Configure NVT Server
SetNetworkInterfacesResponse
(RebootNeeded)
Receive and display the message
SystemReboot (if necessary)
(Empty)
Reboot NVT if necessary.
SystemRebootResponse
(Message)
(Reboot in progress)
Multicast Hello Message
(Reboot successful)
Receive and validate Hello Message
GetNetworkInterfacesRequest
(Empty)
GetNetworkInterfacesResponse
(modified settings)
Return Network Interfaces configuration
Receive and validate the message



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNetworkInterfacesRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetNetworkInterfacesRequest message to set NVT to use Neighbor Discovery for IPv6 stateless IP configuration (InterfaceToken = available network interface, NetworkInterface.IPv6.Enabled = true, NetworkInterface.IPv6.DHCP = "Off").
5. NVT will return SetNetworkInterfacesResponse message.
6. If necessary, NVC will invoke SystemReboot message to restart NVT. Otherwise, continue to step-8.
7. NVT will return SystemRebootResponse message.
8. NVT will send Multicast Hello message from newly configured address.
9. NVC will receive and validate Hello message sent from newly configured address by NVT.
10. NVC will invoke GetNetworkInterfacesRequest message to newly configured address to retrieve the modified settings of NVT.
11. NVC will receive and validate GetNetworkInterfacesResponse message sent from newly configured address by NVT.
12. NVC will restore the original settings by following the procedure mentioned in Annex A.13.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNetworkInterfacesResponse message.

The DUT did not send SystemRebootResponse message.

The DUT did not send Hello message after IP configuration change.

The DUT did not send GetNetworkInterfacesResponse message.

The DUT did not send correct network interface information (i.e. NetworkInterface[token = available network interface token, IPv6.Enabled = true, IPv6.Config.LinkLocal = new IP address]).in GetNetworkInterfacesResponse message.

4.2.4 NVT IPV6 STATEFUL IP CONFIGURATION

Test Label: IP Configuration NVT IPv6 Stateful IP Configuration (DHCPv6)

ONVIF Core Specification Coverage: 6. IP Configuration

Device Type: NVT

Command Under Test: None

WSDL Reference: devicemgmt.wsdl

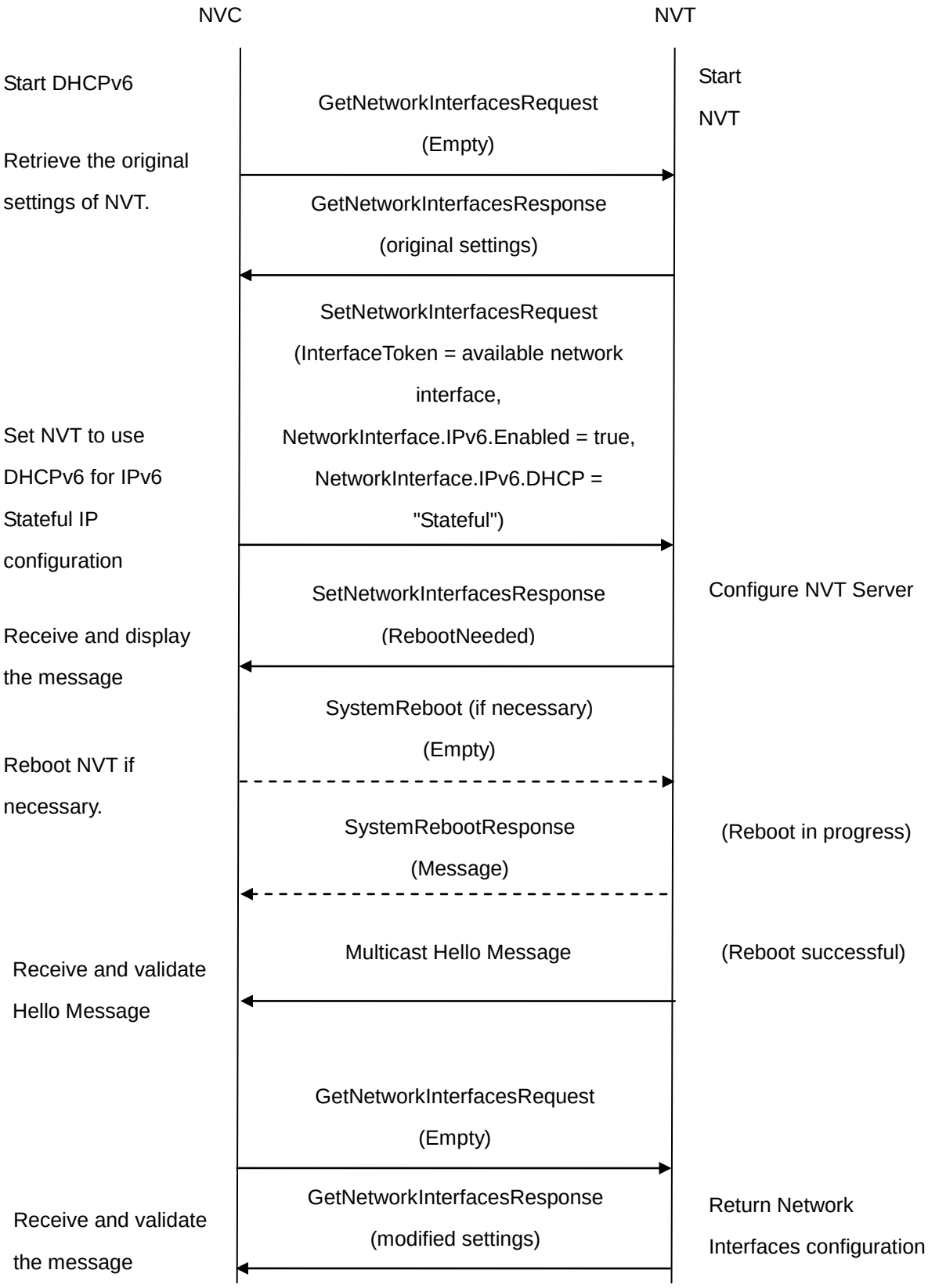
Requirement Level: MUST IF IMPLEMENTED (IPv6)

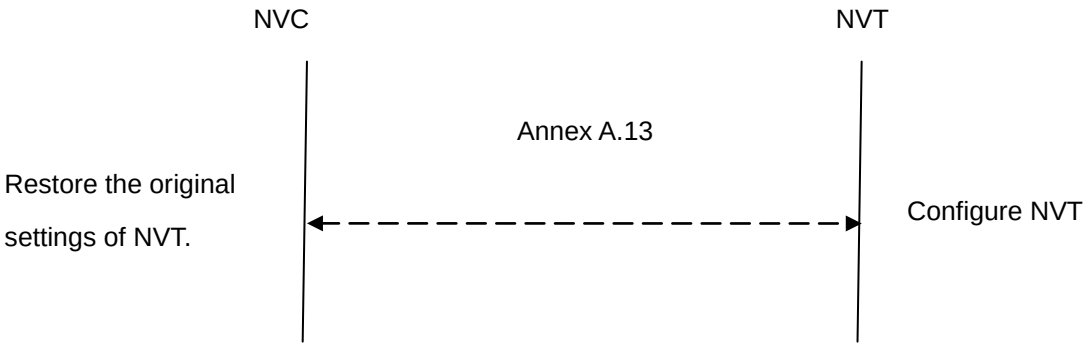
Test Purpose: To test IPv6 Stateful IP Configuration (DHCPv6)

Pre-Requisite: IPv6 is implemented by NVT.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start DHCPv6
Start NVT
GetNetworkInterfacesRequest
(Empty)
Retrieve the original settings of NVT.
GetNetworkInterfacesResponse
(original settings)
SetNetworkInterfacesRequest
(InterfaceToken = available network interface,
NetworkInterface.IPv6.Enabled = true,
NetworkInterface.IPv6.DHCP = "Stateful")
Set NVT to use DHCPv6 for IPv6 Stateful IP configuration
SetNetworkInterfacesResponse
(RebootNeeded)
Configure NVT Server
Receive and display the message
SystemReboot (if necessary)
(Empty)
Reboot NVT if necessary.
SystemRebootResponse
(Message)
(Reboot in progress)
Multicast Hello Message
(Reboot successful)
Receive and validate Hello Message
GetNetworkInterfacesRequest
(Empty)
GetNetworkInterfacesResponse
(modified settings)
Return Network Interfaces configuration
Receive and validate the message



Test Procedure:

1. Start DHCPv6 server.
2. Start an NVC.
3. Start an NVT.
4. NVC will invoke GetNetworkInterfacesRequest message to retrieve the original settings of NVT.
5. NVC will invoke SetNetworkInterfacesRequest message to set NVT to accept Router Advertisement for IPv6 stateful IP configuration (InterfaceToken = available network interface, NetworkInterface.IPv6.Enabled = true, NetworkInterface.IPv6.DHCP = "Stateful").
6. NVT will return SetNetworkInterfacesResponse message.
7. If necessary, NVC will invoke SystemReboot message to restart NVT. Otherwise, continue to step-9.
8. NVT will return SystemRebootResponse message.
9. NVT will send Multicast Hello message from newly configured address.
10. NVC will receive and validate Hello message sent from newly configured address by NVT.
11. NVC will invoke GetNetworkInterfacesRequest message to newly configured address to retrieve the modified settings of NVT.
12. NVC will receive and validate GetNetworkInterfacesResponse message sent from newly configured address by NVT.
13. NVC will restore the original settings by following the procedure mentioned in Annex A.13.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNetworkInterfacesResponse message.

The DUT did not send SystemRebootResponse message.

The DUT did not send Hello message after IP configuration change.

The DUT did not send GetNetworkInterfacesResponse message.

The DUT did not send correct network interface information (i.e. NetworkInterface[token = available network interface token, IPv6.Enabled = true, IPv6.Config.FromDHCP = new IP address]).in GetNetworkInterfacesResponse message.

5 Device Discovery Test Cases

This section covers tests designed for NVT Device Discovery Feature.

5.1 NVT HELLO MESSAGE

Test Label: Device Discovery NVT Multicast HELLO Message Transmission.

ONVIF Core Specification Coverage: 7.3.3 Hello, 8.3.10 Reboot

Device Type: NVT

Command Under Test: Hello

WSDL Reference: ws-discovery.wsdl, devicemgmt.wsdl

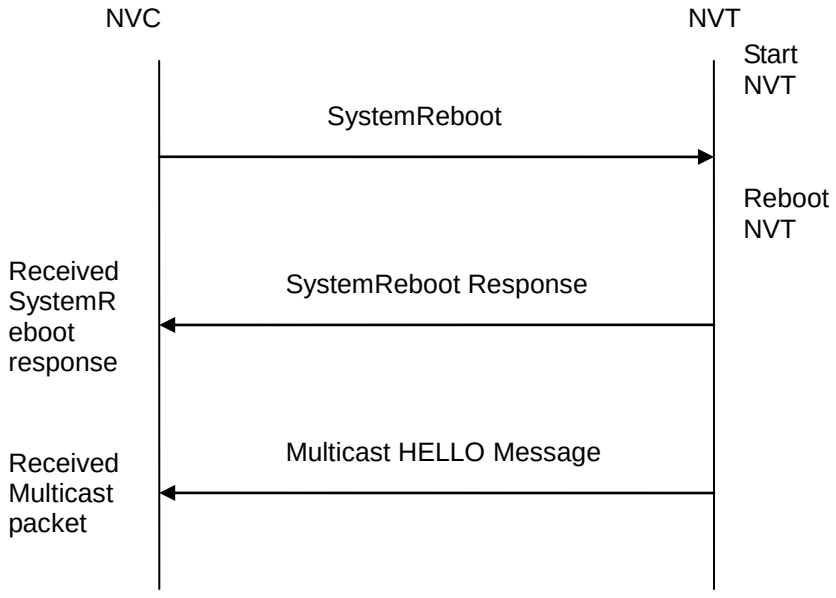
Requirement Level: MUST

Test Purpose: To verify that the NVT transmits HELLO message with the correct multicast parameters (address, and port number) when it is connected to the network.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC
2. Start an NVT
3. NVC invokes SystemReboot message to reboot the NVT.
4. NVT sends SystemRebootResponse message.

5. NVC waits for the user defined boot time to receive HELLO message from NVT.
6. Verify that the NVT transmits the HELLO message with multicast address 239.255.255.250, and port number 3702.
7. In case of device discovery with IPv6 interface, verify that the NVT transmits the HELLO message with multicast address FF02::C(link-local scope),and port number 3702.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SystemRebootResponse message.

The DUT did not send the multicast HELLO message.

5.2 NVT HELLO MESSAGE VALIDATION

Test Label: Device Discovery NVT HELLO Message Validation

ONVIF Core Specification Coverage: 7.3.1 Endpoint reference, 7.3.3 Hello, 7.3.3.1 Types, 7.3.3.2 Scopes, 8.3.10 Reboot

Device Type: NVT

Command Under Test: Hello

WSDL Reference: ws-discovery.wsdl, devicemgmt.wsdl

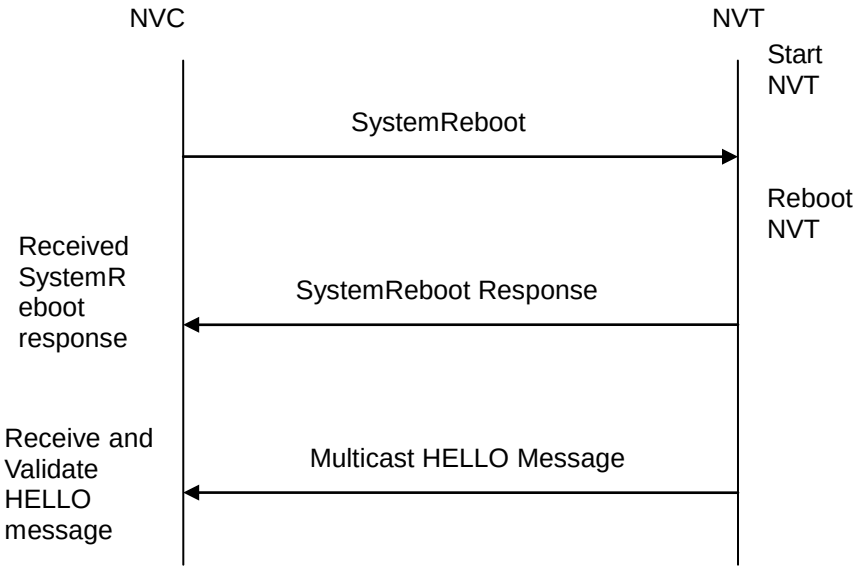
Requirement Level: MUST

Test Purpose: To verify the mandatory XML elements Device type, Scope types, Endpoint Reference and Meta data version in the HELLO message.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes SystemReboot message to reboot the NVT.
4. NVT sends SystemRebootResponse message.
5. NVC waits for the user defined boot time to receive HELLO message from NVT.
6. NVC will verify the mandatory XML elements in the NVT HELLO message.

Test Result:

PASS –

DUT passes all assertions

FAIL –

The DUT did not send SystemRebootResponse message.

The DUT did not send multicast HELLO message.

The DUT did not send HELLO message with one or more mandatory XML elements (EndpointReference, Types, and Scopes).

The DUT did not send HELLO message with mandatory device type and scope types (type, location, hardware and name).

The DUT did not send HELLO message with a namespace of the Types value ("http://www.onvif.org/ver10/network/wsdl").

Note: See Annex A.1 for Device and Scope Types definition.

5.3 NVT SEARCH BASED ON DEVICE SCOPE TYPES

Test Label: Device Discovery NVT Search based on device scope types.

ONVIF Core Specification Coverage: 5.1 Services overview, 7.3.3.1 Types, 7.3.3.2 Scopes, 7.3.4 Probe and Probe Match, 8.3.11 Get scope parameters

Device Type: NVT

Command Under Test: Probe, ProbeMatch

WSDL Reference: ws-discovery.wsdl, devicemgmt.wsdl

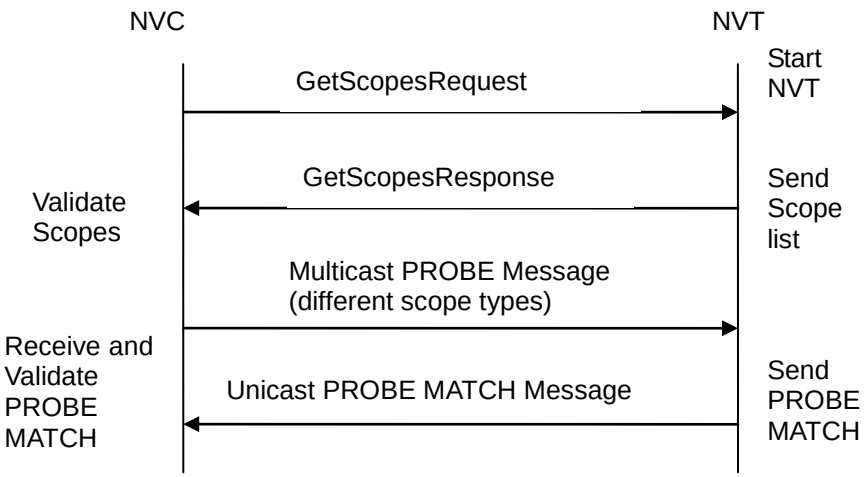
Requirement Level: MUST

Test Purpose: To search the NVT based on the mandatory scope types (type, location, hardware and name).

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT
3. NVC will invoke GetScopesRequest message to retrieve existing scopes list.
4. NVT replies with the list of scopes types in GetScopesResponse message.
5. NVC will transmit the multicast PROBE message with different scope types (type, location, hardware and name).
6. NVC will verify the PROBE MATCH message sent by NVT.

Test Result:

PASS –

DUT passes all assertions

FAIL –

The DUT did not send GetScopesResponse message.

The DUT scope list does not have one or more mandatory scope entry.

The DUT did not send mandatory XML elements (device, scope type, service address and scope matching rule) in the PROBE MATCH message.

The DUT did not send PROBE MATCH message within the time out period of APP_MAX_DELAY

The DUT did not send PROBE MATCH message with a namespace of the Types value ("http://www.onvif.org/ver10/network/wsdl").

The DUT did not send PROBE MATCH message with fixed entry point to <d:XAddrs> element (http://onvif_host/onvif/device_service).

In case of IPv6, the DUT did not send PROBE MATCH message with <d:XAddrs> of including IPv6 address.

The DUT did not send PROBE MATCH message with a correct XML element RelatesTo that it has same value as MessageID of PROBE message (not to omit "urn" namespace).

Note: See Annex A.1 for Device and Scope Types definition, Annex A.12 for the value of APP_MAX_DELAY.

5.4 NVT SEARCH WITH OMITTED DEVICE AND SCOPE TYPES

Test Label: Device Discovery NVT Search with omitted device type and scope types.

ONVIF Core Specification Coverage: 5.1 Services overview, 7.3.3.1 Types, 7.3.3.2 Scopes, 7.3.4 Probe and Probe Match

Device Type: NVT

Command Under Test: Probe, ProbeMatch

WSDL Reference: ws-discovery.wsdl

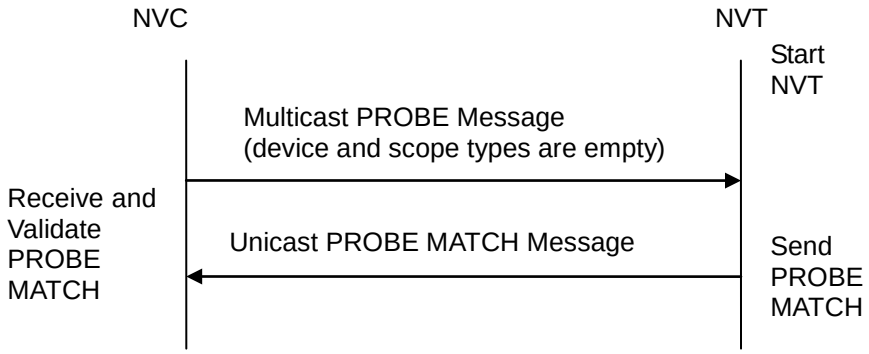
Requirement Level: MUST

Test Purpose: To search the NVT with device and scope types being omitted.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will transmit multicast PROBE message with device type and scope type inputs omitted.
4. NVC will verify the PROBE MATCH message sent by NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send PROBE MATCH message within the time out period of APP_MAX_DELAY

The DUT did not send mandatory XML elements (device, scope type, service address and scope matching rule) in the PROBE MATCH message.

The DUT did not send PROBE MATCH message with a namespace of the Types value ("http://www.onvif.org/ver10/network/wsdl").

The DUT did not send PROBE MATCH message with fixed entry point to <d:XAddrs> element (http://onvif_host/onvif/device_service).

In case of IPv6, the DUT did not send PROBE MATCH message with <d:XAddrs> of including IPv6 address.

The DUT did not send PROBE MATCH message with a correct XML element RelatesTo that it has same value as MessageID of PROBE message (not to omit "urn" namespace).

Note: See Annex A.12 for the value of APP_MAX_DELAY.

5.5 NVT RESPONSE TO INVALID SEARCH REQUEST

Test Label: Device Discovery NVT does not respond to invalid multicast PROBE message.

ONVIF Core Specification Coverage: 7.3.4 Probe and Probe Match

Device Type: NVT

Command Under Test: Probe

WSDL Reference: ws-discovery.wsdl

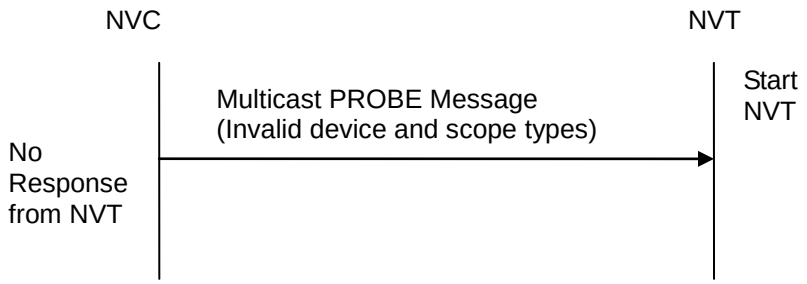
Requirement Level: MUST

Test Purpose: To verify that NVT do not reply to the invalid multicast PROBE message (invalid device and scope types).

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will transmit multicast PROBE message with invalid device and scope types.
4. Verify that the NVT did not send PROBE MATCH message.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did send PROBE MATCH message.

Note: See Annex A.1 for Invalid Device and Scope Types definition.

5.6 NVT SEARCH USING UNICAST PROBE MESSAGE

Test Label: Device Discovery NVT Search by Unicast PROBE message.

Requirement Level: OPTIONAL

Test Purpose: To verify NVT behaviour for Unicast PROBE message.

Note: All Tests 5.3, 5.4, 5.5 to be repeated with Unicast PROBE message.

5.7 NVT DEVICE SCOPES CONFIGURATION

Test Label: Device Discovery NVT Device Scope configurations.

ONVIF Core Specification Coverage: 7.3.3 Hello, 7.3.4 Probe and Probe Match, 8.3.11 Get scope parameters, 8.3.12 Set scope parameters, 8.3.13 Add scope parameters, 8.3.14 Delete scope parameters

Device Type: NVT

Command Under Test: AddScopes, SetScopes, RemoveScopes

WSDL Reference: ws-discovery.wsdl, devicemgmt.wsdl

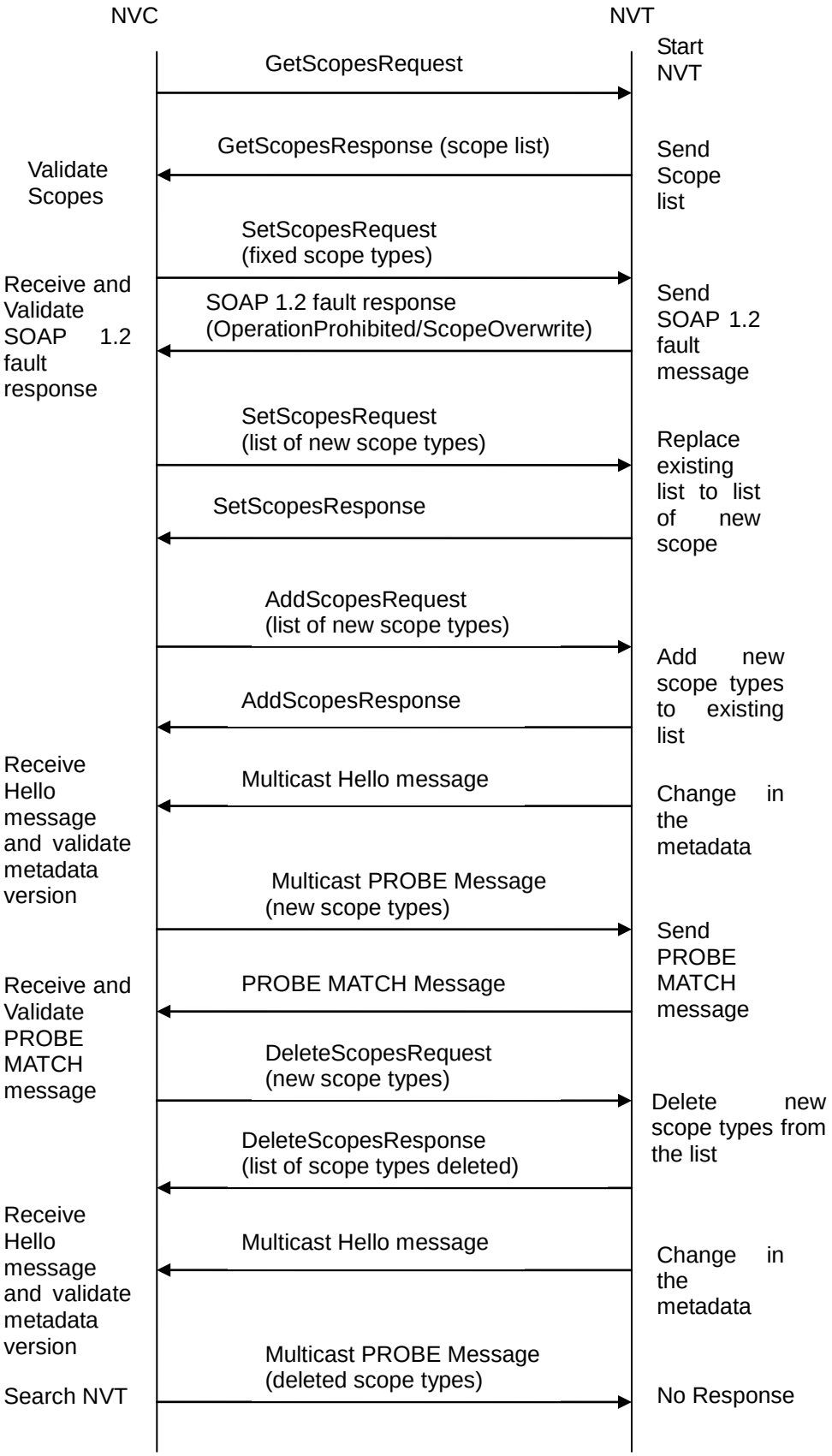
Requirement Level: MUST

Test Purpose: To verify NVT behaviour for scope parameter configuration.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
GetScopesRequest
Start NVT
GetScopesResponse (scope list)
Send Scope list
Validate Scopes
SetScopesRequest (fixed scope types)
Receive and Validate SOAP 1.2 fault response
SOAP 1.2 fault response (OperationProhibited/ScopeOverwrite)
Send SOAP 1.2 fault message
SetScopesRequest (list of new scope types)
Replace existing list to list of new scope
SetScopesResponse
AddScopesRequest (list of new scope types)
Add new scope types to existing list
AddScopesResponse
Receive Hello message and validate metadata version
Multicast Hello message
Change in the metadata
Multicast PROBE Message (new scope types)
Send PROBE MATCH message
Receive and Validate PROBE MATCH message
PROBE MATCH Message
DeleteScopesRequest (new scope types)
Delete new scope types from the list
DeleteScopesResponse (list of scope types deleted)
Receive Hello message and validate metadata version
Multicast Hello message
Change in the metadata
Search NVT
Multicast PROBE Message (deleted scope types)
No Response

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetScopesRequest message to retrieve existing scope types.
4. NVT replies with the list of scopes types in the GetScopesResponse message.
5. NVC will invoke SetScopesRequest message to overwrite existing scope types with new scope types.
6. NVT generates SOAP 1.2 fault message as fixed scope types cannot be overwritten.
7. NVC will invoke SetScopesRequest message to replace existing list to list of new scope.
8. NVT replies with SetScopesResponse message indicating success.
9. NVC will invoke AddScopesRequest message to add new scope types to the existing scope list.
10. NVT replies with AddScopesResponse message indicating success.
11. NVT sends Multicast Hello message to indicate the change in the metadata (i.e. addition of new scope types to the existing list).
12. NVC will invoke Multicast PROBE message to search NVT with newly added scope types.
13. Verify that NVT issued a PROBE MATCH message.
14. NVC will invoke DeleteScopesRequest message to delete the newly configured scope types.
15. NVT replies with DeleteScopesResponse message indicating success.
16. NVT sends Multicast Hello message to indicate the change in the metadata (i.e. deletion of scope types from the existing list).
17. NVC will invoke Multicast PROBE message to search NVT with deleted scope types.
18. Verify that the NVT did not send PROBE MATCH message.

Test Result:

PASS –

DUT passes all assertions

FAIL –

The DUT did not send GetScopesResponse message.

The DUT scope list does not have one or more mandatory scope entry.

The DUT did not send SOAP 1.2 fault message (OperationProhibited/ScopeOverwrite) after executing Test Procedure 5.

The DUT did not send SetScopesResponse message after executing Test Procedure 7.

The DUT did not send AddScopesResponse message.

The DUT did not send multicast Hello message after the change in its metadata (addition/deletion of scope types).

The DUT did not send mandatory XML elements (device, new scope type, service address and scope matching rule) in the PROBE MATCH message.

The DUT did not send DeleteScopesResponse message.

The DUT did not send PROBE MATCH message within the time out period of APP_MAX_DELAY after executing Test Procedure steps 12 and 17.

Note:

It may be possible that NVT may return SOAP Fault 1.2 “TooManyScopes” for the SetScopes (step 7) or AddScopes (step 9) command. Such SOAP 1.2 fault message shall be treated as PASS case for this test.

Whenever there is a change in the metadata of the Target Service, “**MetadataVersion**” is incremented by >=1.

See Annex A.6 for Invalid SOAP 1.2 fault message definition and Annex A.12 for the value of APP_MAX_DELAY.

5.8 NVT BYE MESSAGE

Test Label: Device Discovery NVT BYE Message Transmission.

ONVIF Core Specification Coverage: 7.3.6 Bye, 8.3.10 Reboot

Device Type: NVT

Command Under Test: Bye

WSDL Reference: ws-discovery.wsdl, devicemgmt.wsdl

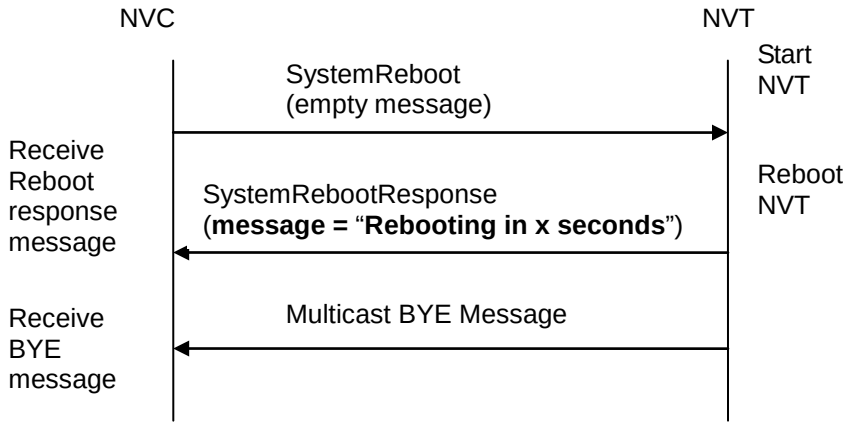
Requirement Level: SHOULD

Test Purpose: To verify that NVT transmits BYE message before the system reboot.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SystemReboot message to reboot the NVT.
4. Verify that NVT sends SystemRebootResponse message (example message string = "Rebooting in x seconds").
5. Verify that the NVT issued a BYE message.
6. NVC waits for the user defined boot time before proceeding to execute next test case.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SystemRebootResponse message.

The DUT did not send BYE message.

5.9 NVT DISCOVERY MODE CONFIGURATION

Test Label: Device Discovery NVT Discovery mode configurations.

ONVIF Core Specification Coverage: 7.3.4 Probe and Probe Match, 8.3.10 Reboot, 8.3.15 Get discovery mode, 8.3.16 Set discovery mode

Device Type: NVT

Command Under Test: GetDiscoveryMode, SetDiscoveryMode

WSDL Reference: ws-discovery.wsdl, devicemgmt.wsdl

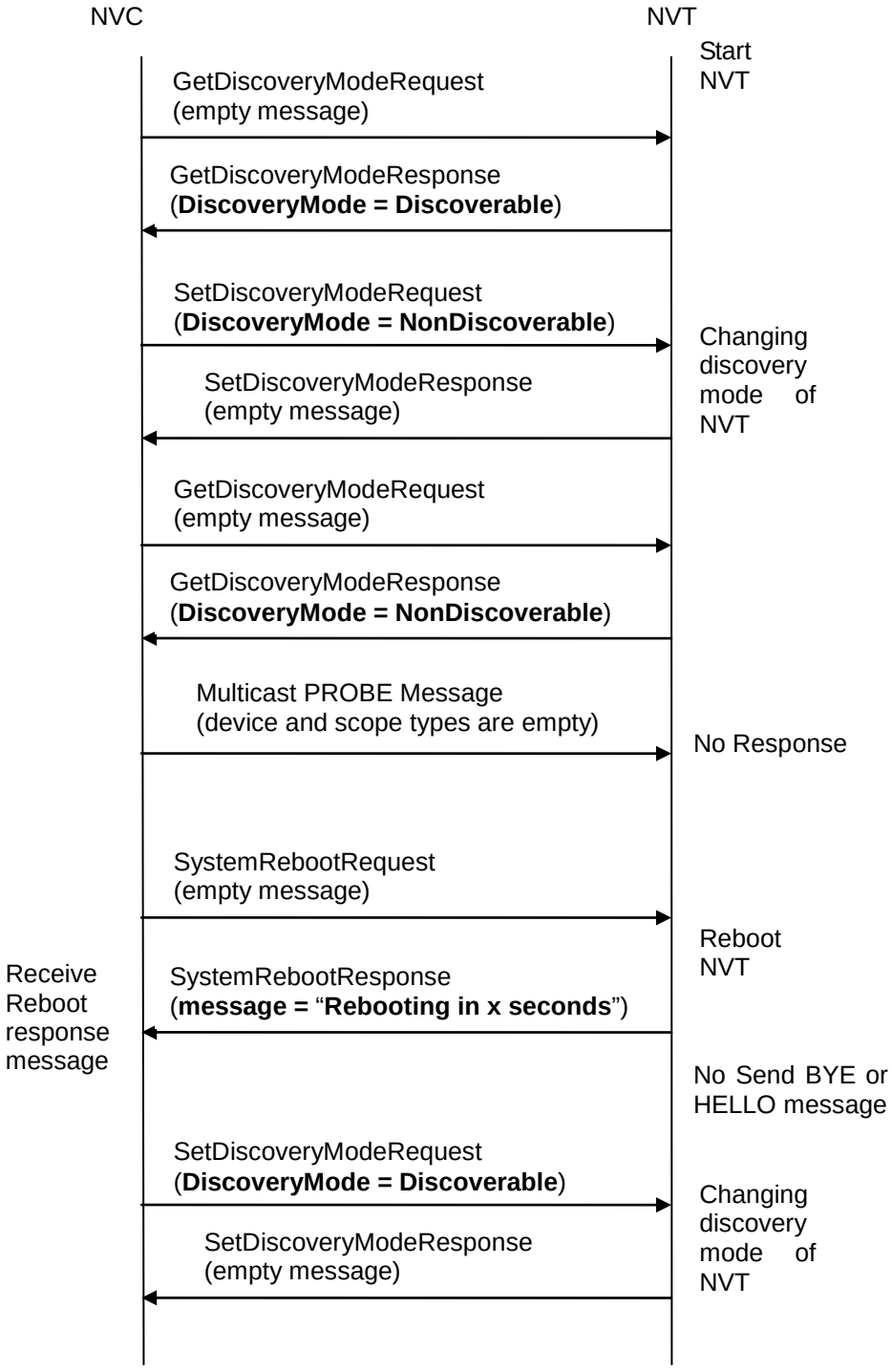
Requirement Level: MUST

Test Purpose: To verify NVT behaviour for Discovery mode configuration.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.

2. Start an NVT.
3. NVC will invoke GetDiscoveryMode message to verify discovery mode of the NVT.
4. Verify that NVT sends GetDiscoveryModeResponse message (DiscoveryMode DiscoveryMode = Discoverable).
5. NVC will invoke SetDiscoveryMode message to set discovery mode of the NVT to Non-Discoverable.
6. Verify that NVT sends SetDiscoveryModeResponse message.
7. NVC will invoke GetDiscoveryMode message to verify discovery mode of the NVT.
8. Verify that NVT sends GetDiscoveryModeResponse message (DiscoveryMode DiscoveryMode = NonDiscoverable).
9. NVC will transmit multicast PROBE message with device type and scope type inputs omitted.
10. Verify that the NVT did not send PROBE MATCH message.
11. NVC will invoke SystemReboot message to reboot the NVT.
12. Verify that NVT sends SystemRebootResponse message (example message string = “Rebooting in x seconds”).
13. Verify that the NVT did not send BYE or HELLO message.
14. NVC waits for the user defined boot time before proceeding to execute next step.
15. NVC will invoke SetDiscoveryMode message to set discovery mode of the NVT to Discoverable.
16. Verify that NVT sends SetDiscoveryModeResponse message.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetDiscoveryModeResponse message.

The DUT GetDiscoveryModeResponse did not have a Discovery mode parameter of Discoverable after executing Test Procedure 3.

The DUT did not send SetDiscoveryModeResponse message.

The DUT GetDiscoveryModeResponse did not have a Discovery mode parameter of NonDiscoverable after executing Test Procedure 7.

The DUT sent PROBE MATCH message within the time out period of APP_MAX_DELAY

The DUT did not send SystemRebootResponse message.

The DUT sent BYE or HELLO message after executing Test Procedure 12.

Note: See Annex A.12 for the value of APP_MAX_DELAY.

5.10 NVT SOAP FAULT MESSAGE

Test Label: Device Discovery NVT generates SOAP 1.2 fault message for Invalid Unicast PROBE Message.

ONVIF Core Specification Coverage: 7.3.7 SOAP Fault Messages

Device Type: NVT

Command Under Test: Probe

WSDL Reference: ws-discovery.wsdl, devicemgmt.wsdl

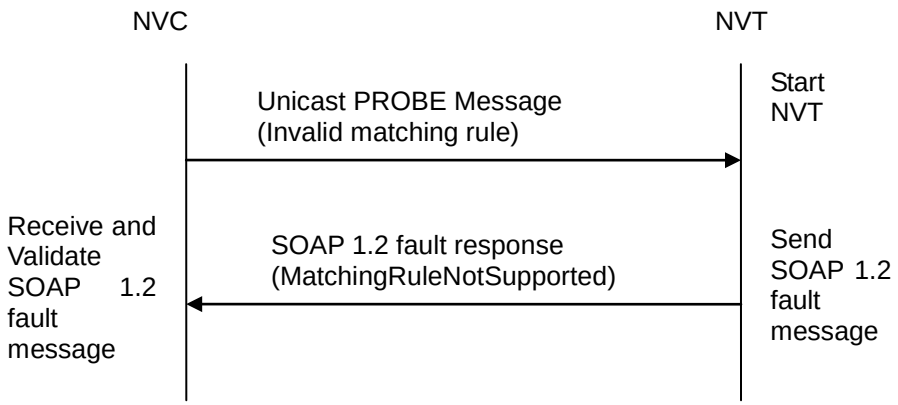
Requirement Level: OPTIONAL

Test Purpose: To verify that NVT generates a SOAP 1.2 fault message to the invalid Unicast PROBE message (Invalid matching rule).

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will transmit Unicast PROBE message with invalid matching type rule.
4. Verify that the NVT generates a SOAP 1.2 fault message (MatchingRuleNotSupported).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct SOAP 1.2 fault message (fault code, namespace etc).

Note: See Annex A.6 for Invalid SOAP 1.2 fault message definition. Refer RFC 3986 for scope matching definitions.

6 Device Management Test Cases

6.1 Capabilities

6.1.1 NVT GET WSDL URL

Test Label: Device Management NVT WSDL URL.

ONVIF Core Specification Coverage: 8.1.1 Get WSDL URL

Command under test: GetWsdlUrl

Device Type: NVT

Requirement Level: MUST

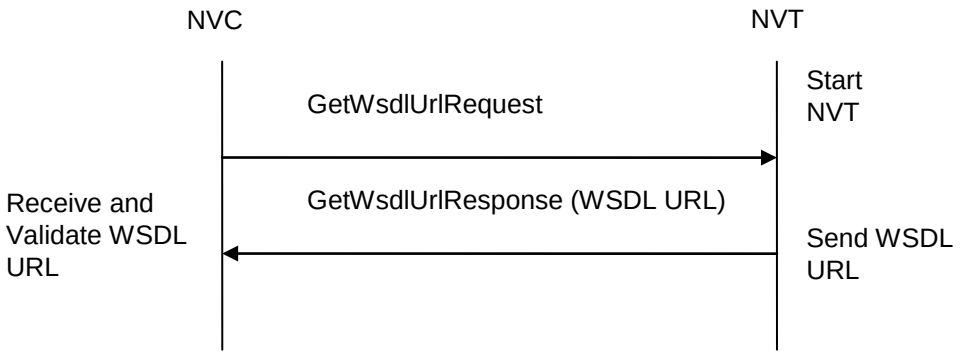
WSDL Reference: devicemgmt.wsdl

Test Purpose: To retrieve complete XML schema and WSDL definitions of the NVT.

Pre-Requisite: None.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetWsdlUrlRequest message to retrieve XML schema and WSDL definitions of the NVT.
4. Verify that NVT sends GetWsdlUrlResponse message (WSDL URL).
5. Validate the WSDL URL returned from the NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetWsdlUrlResponse message.

6.1.2 NVT ALL CAPABILITIES

Test Label: Device Management All Capabilities Verification.

ONVIF Core Specification Coverage: 8.1.2 Capability exchange

Device Type: NVT

Command under test: GetCapabilities

Requirement Level: MUST

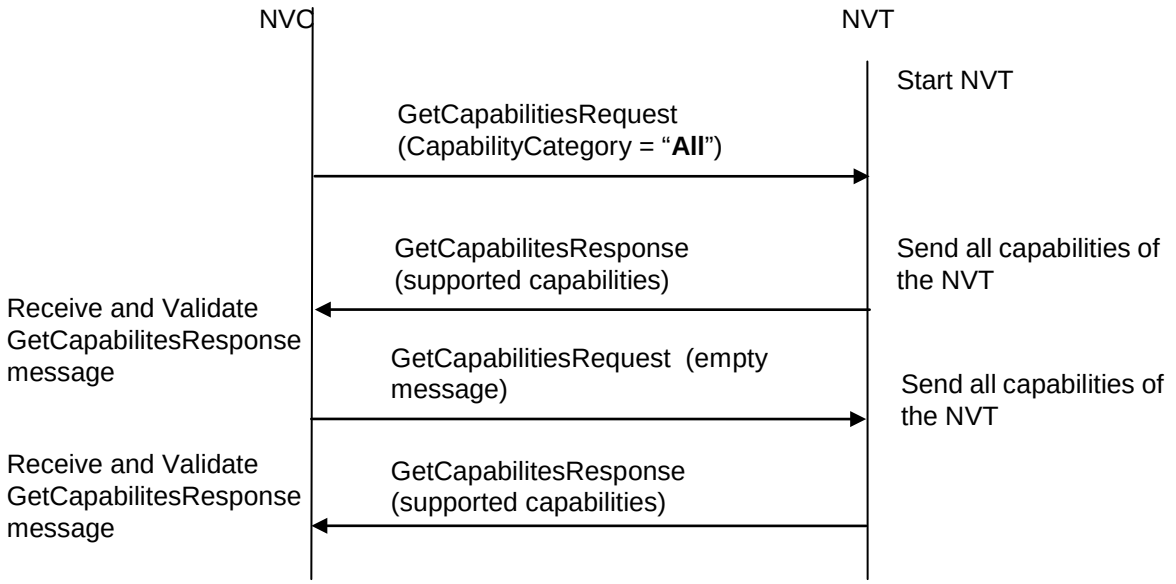
WSDL Reference: devicemgmt.wsdl

Test Purpose: To verify all Capabilities of the NVT.

Pre-Requisite: None.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.

3. NVC will invoke GetCapabilitiesRequest message (CapabilityCategory = “All”) to retrieve all capabilities of the NVT.
4. Verify the Capabilities Response from the NVT and support for Device, Media and Events capabilities.
5. NVC will invoke GetCapabilitiesRequest message (empty message) to retrieve all capabilities of the NVT.
6. Verify the Capabilities Response from the NVT and support for Device, Media and Events capabilities

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetCapabilitesResponse message at step 4 and step 5.

The DUT did not support Device, Media and Events capabilities.

6.1.3 NVT DEVICE CAPABILITIES

Test Label: Device Management NVT Device Capabilities Verification.

ONVIF Core Specification Coverage: 8.1.2 Capability exchange.

Device Type: NVT

Command under test: GetCapabilities.

Requirement Level: MUST

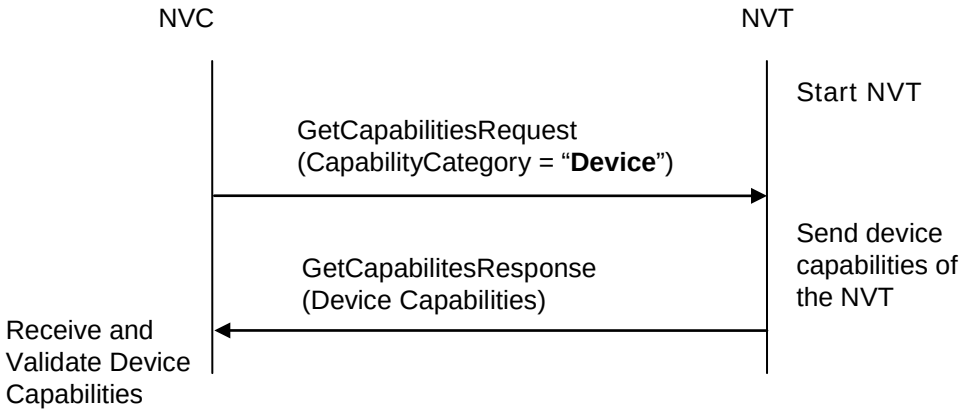
WSDL Reference: devicemgmt.wsdl

Test Purpose: To verify Device Capabilities of the NVT.

Pre-Requisite: None.

Test Configuration: NVC and NVT.

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetCapabilitiesRequest message (CapabilityCategory = “Device”) to retrieve Device Capabilities of the NVT.
4. NVT sends its device capabilities in the GetCapabilitesResponse message.
5. Verify the address of the device service in the GetCapabilitesResponse message.
6. Verify Network, System, IO and Security capabilities if supported by the DUT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetCapabilitesResponse message.

The DUT did not send the address of the device service.

6.1.4 NVT MEDIA CAPABILITIES

Test Label: Device Management NVT Media Capabilities Verification.

ONVIF Core Specification Coverage: 8.1.2 Capability exchange

Device Type: NVT

Command under test: GetCapabilities

Requirement Level: MUST

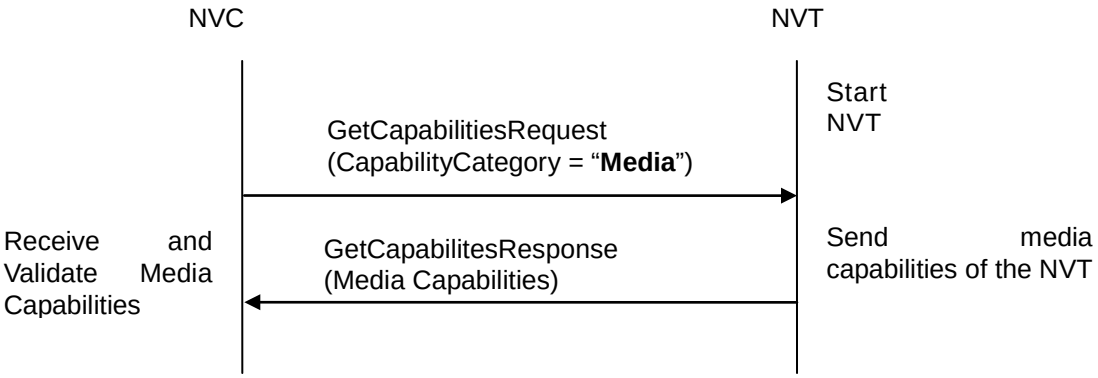
WSDL Reference: devicemgmt.wsdl

Test Purpose: To verify Media Capabilities of the NVT.

Pre-Requisite: None.

Test Configuration: NVC and NVT.

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetCapabilitiesRequest message (CapabilityCategory = "Media") to retrieve Media Capabilities of the NVT.
4. NVT sends its media capabilities in the GetCapabilitesResponse message.
5. Verify the address of the media service in the GetCapabilitesResponse message.
6. Verify Real time streaming capabilities if supported by the DUT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetCapabilitesResponse message.

The DUT did not send the address of the media service.

6.1.5 NVT EVENT CAPABALITIES

Test Label: Device Management NVT Event Capabilities Verification.

ONVIF Core Specification Coverage: 8.1.2 Capability exchange.

Device Type: NVT.

Command Under Test: GetCapabilities.

WSDL Reference: devicemgmt.wsdl

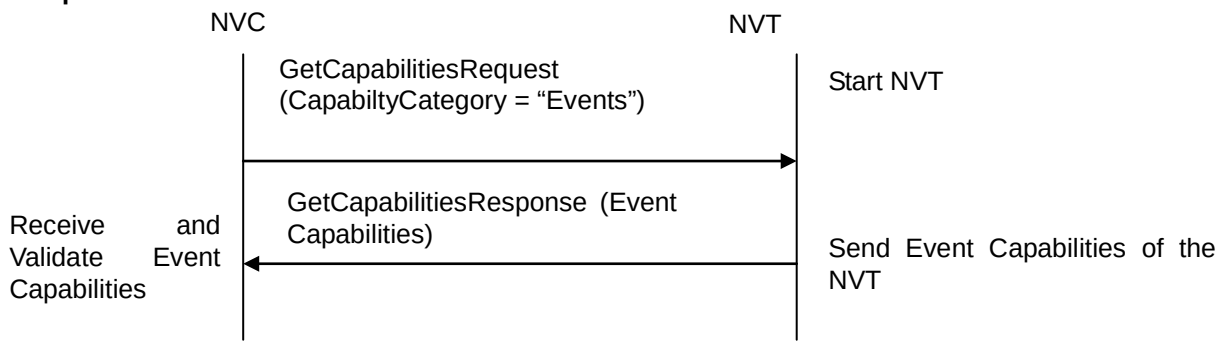
Requirement Level: MUST.

Test Purpose: To verify event capabilities of the NVT.

Pre-Requisite: None.

Test Configuration: NVC and NVT.

Test Sequence:



Test Procedure:

1. Start an NVT.
2. Start an NVC.
3. NVC will invoke GetCapabilitiesRequest message (CapabilityCategory = “Events”) to retrieve Event Capabilities of the NVT.
4. Verify the address of the event service in the GetCapabilitesResponse message.
5. Verify the Subscription policies if supported by the NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetCapabilitesResponse message.

The DUT did not send the address of the event service.

6.1.6 NVT PTZ CAPABALITIES

Test Label: Device Management NVT PTZ Capabilities Verification.

ONVIF Core Specification Coverage: 8.1.2 Capability exchange.

Device Type: NVT

Command Under Test: GetCapabilities.

WSDL Reference: devicemgmt.wsdl

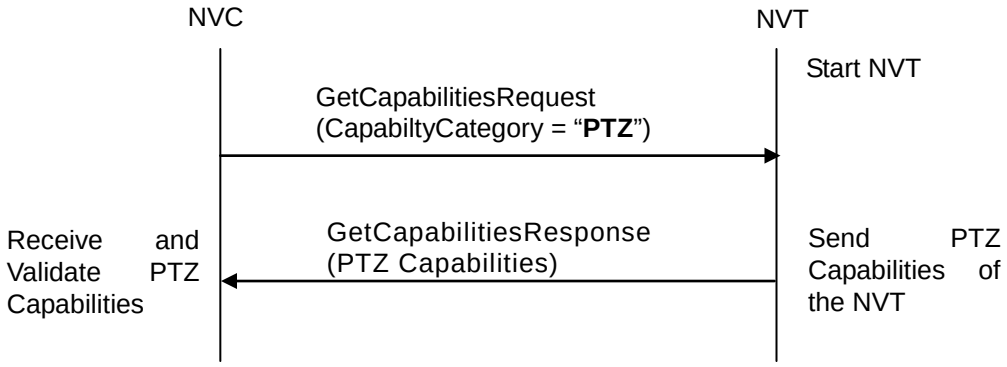
Requirement Level: MUST

Test Purpose: To verify PTZ capabilities of the NVT.

Pre-Requisite: None

Test Configuration: NVC and NVT.

Test Sequence:



Test Procedure:

1. Start an NVT.
2. Start an NVC.
3. NVC will invoke GetCapabilitiesRequest message (CapabilityCategory = "PTZ") to retrieve PTZ Capabilities of the NVT.
4. Verify the GetCapabilitesResponse, message should either have address of the PTZ service or SOAP 1.2 fault message, if the PTZ service is not supported.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetCapabilitesResponse message.

The DUT did not send the address of the PTZ service, if PTZ supported or DUT did not generate the SOAP 1.2 fault message, if the PTZ is not supported.

6.1.7 NVT SERVICE CATEGORY CAPABILITIES

Note: NVT Service Category Capabilities Test to be repeated for Analytics and Imaging service

If a specific service category is not supported by the DUT, it MUST generate SOAP 1.2 fault response (ActionNotSupported/NoSuchService).

6.1.8 NVT SOAP FAULT MESSAGE

Test Label: Device Management NVT generates a SOAP 1.2 fault message for Invalid GetCapabilitiesRequest Message.

Command under test: GetCapabilities.

ONVIF Core Specification Coverage: 8.1.2 Capability exchange.

Device Type: NVT

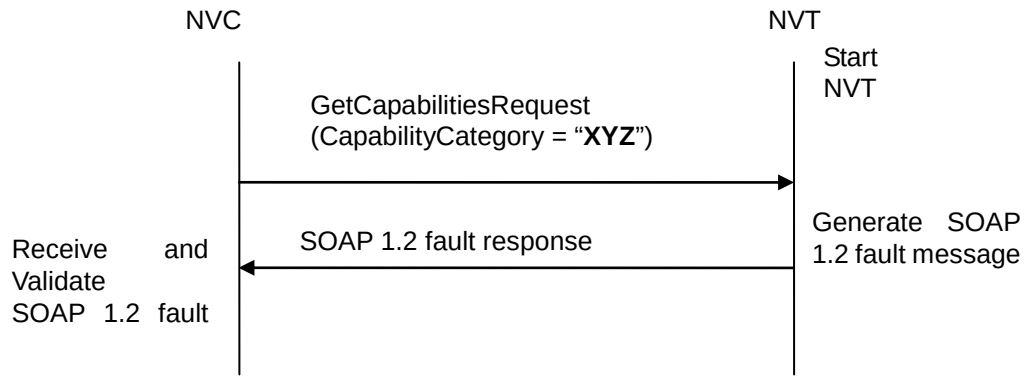
Requirement Level: SHOULD.

WSDL Reference: devicemgmt.wsdl

Pre-Requisite: None

Test Purpose: To verify that NVT generates SOAP 1.2 fault message to the invalid GetCapabilitiesRequest message (invalid capability category).

Test Configuration: NVC and NVT.



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will send GetCapabilitiesRequest message with invalid capability category.
4. Verify that the NVT generates a SOAP 1.2 fault message.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct SOAP 1.2 fault message (fault code, namespace etc).

Note: See Annex A.6 for Invalid SOAP 1.2 fault message definition.

6.2 Network

6.2.1 NVT NETWORK COMMAND HOSTNAME CONFIGURATION

Test Label: Device Management NVT Network Command **GetHostname** Test.

ONVIF Core Specification Coverage: 8.2.1 Get hostname

Device Type: NVT

Command Under Test: GetHostname

Requirement Level: MUST

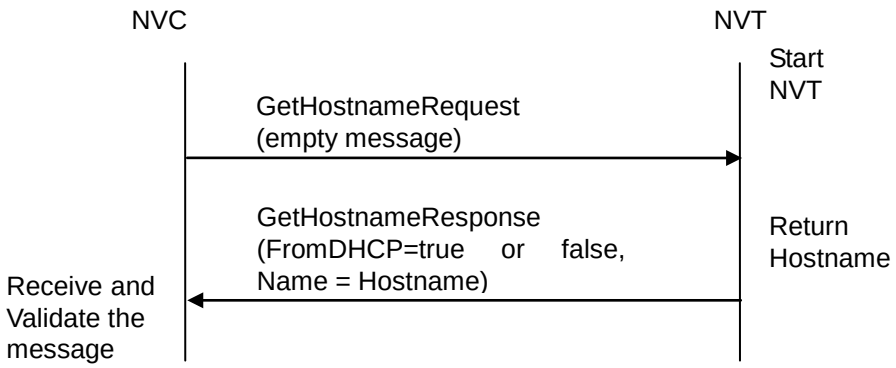
WSDL Reference: devicemgmt.wsdl

Test Purpose: To retrieve hostname of the NVT through **GetHostname** command.

Pre-Requisite: None.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetHostnameRequest (empty message) message to retrieve Hostname of the NVT.
4. Verify the GetHostnameResponse from NVT (FromDHCP = true or false, Name = Hostname).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetHostnameResponse message.

6.2.2 NVT NETWORK COMMAND SETHOSTNAME TEST

Test Label: Device Management NVT Network Command **SetHostname** Test.

ONVIF Core Specification Coverage: 8.2.2 Set hostname

Device Type: NVT

Command Under Test: SetHostname

Requirement Level: MUST

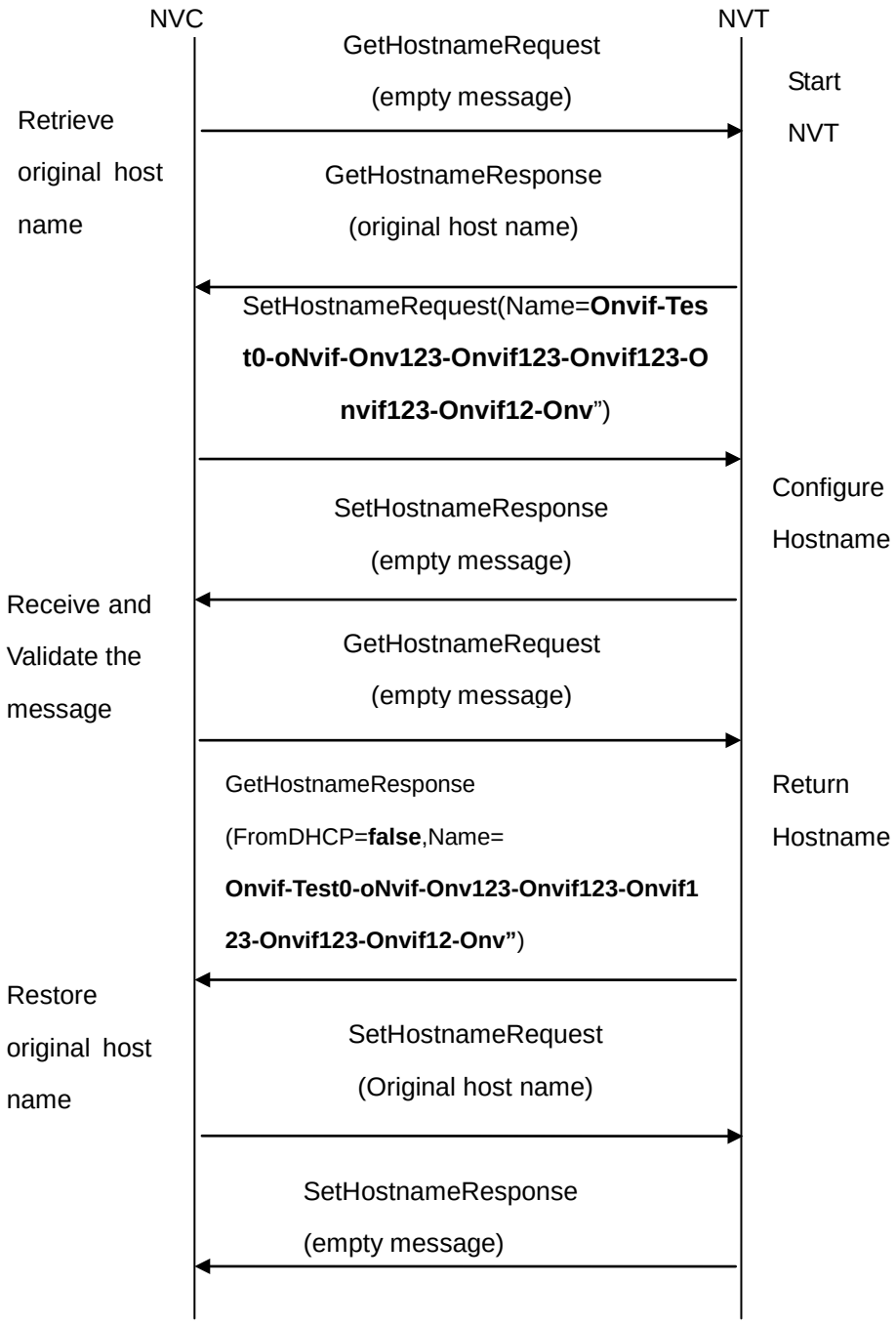
WSDL Reference: devicemgmt.wsdl

Test Purpose: To configure hostname on the NVT through **SetHostname** command.

Pre-Requisite: Testing environment (DHCP server) should not change the IP address of NVT during this test case execution.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes GetHostnameRequest to retrieve original settings of NVT.

4. NVC will invoke SetHostnameRequest message (Name =" Onvif-Test0-oNvif-Onv123-Onvif123-Onvif123-Onvif123-Onvif12-Onv", whose length is equal to 63 bytes) to configure the hostname.
5. Verify that NVT sends SetHostnameResponse (empty message).
6. Verify the hostname configurations in NVT through GetHostnameRequest.
7. NVT sends hostname configuration in the GetHostnameResponse message (FromDHCP = false, Name = "Onvif-Test0-oNvif-Onv123-Onvif123-Onvif123-Onvif123-Onvif12-Onv").

13. NVC invokes SetHostnameRequest to restore original settings of NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetHostnameResponse message.

The DUT did not send GetHostnameResponse message.

The DUT did not send correct hostname (i.e. "Onvif-Test0-oNvif-Onv123-Onvif123-Onvif123-Onvif123-Onvif12-Onv") in the GetHostnameResponse message.

Note: See Annex A.2 for valid host names.

6.2.3 NVT NETWORK COMMAND SETHOSTNAME TEST ERROR CASE

Test Label: Device Management NVT Network Command **SetHostname** Test for invalid hostname.

ONVIF Core Specification Coverage: 8.2.2 Set hostname

Device Type: NVT

Command Under Test: SetHostName.

Requirement Level: SHOULD

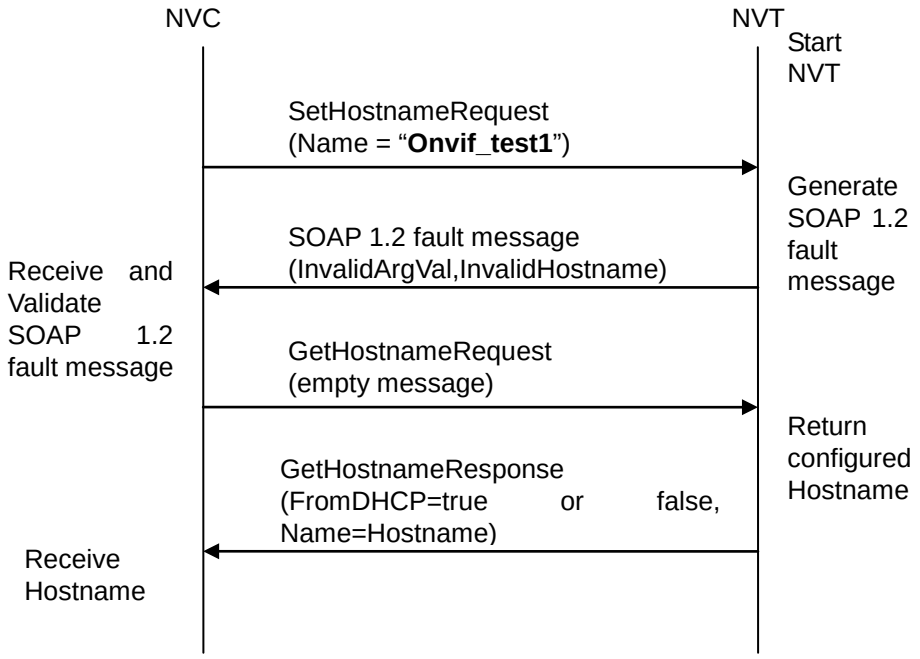
WSDL Reference: devicemgmt.wsdl

Test Purpose: To verify behaviour of NVT for invalid hostname configuration.

Pre-Requisite: Testing environment (DHCP server) should not change the IP address of NVT during this test case execution.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetHostnameRequest message (Name = "Onvif_test1") to configure the hostname.
4. Verify that NVT generates SOAP 1.2 fault message (InvalidArgVal/InvalidHostname).
5. Verify hostname from NVT through GetHostnameRequest.
6. NVT sends valid hostname in GetHostnameResponse message

   (FromDHCP=true or false, Name=Hostname).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct fault code in the SOAP 1.2 fault message (InvalidArgVal, InvalidHostname).

The DUT did not send GetHostnameResponse message.

The DUT returned "Onvif_test1" as its Hostname.

Note: Hostname “Onvif_test1” is just an example. See Annex A.2 for Invalid Hostname and SOAP 1.2 fault message definitions.

6.2.4 NVT GET DNS CONFIGURATION

Test Label: Device Management NVT Network Command GetDNS Test.

ONVIF Core Specification Coverage: 8.2.3 Get DNS settings

Device Type: NVT

Command Under Test: GetDNS

WSDL Reference: devicemgmt.wsdl

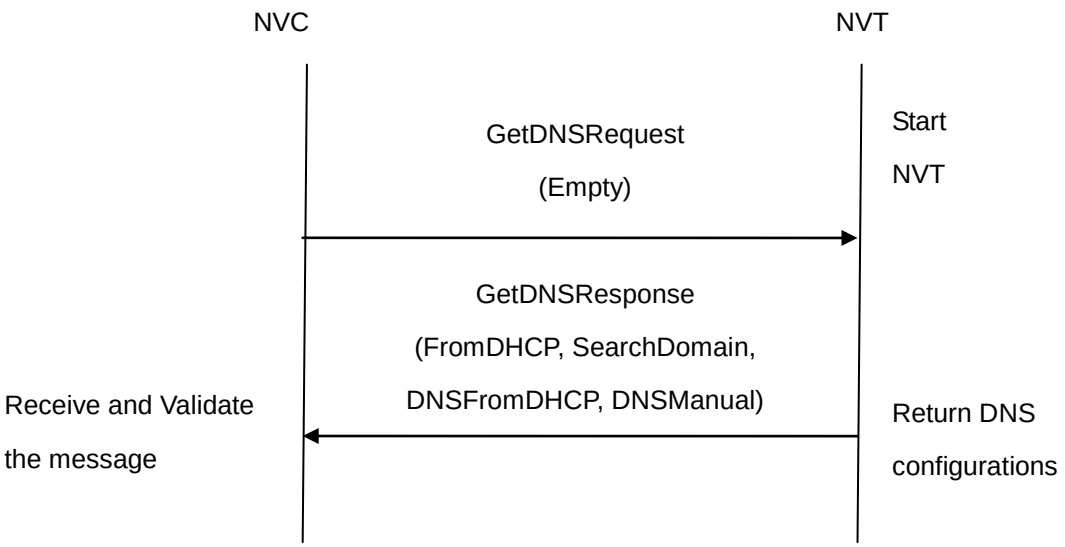
Requirement Level: MUST

Test Purpose: To retrieve DNS configurations of NVT through GetDNS command.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetDNSRequest message to retrieve DNS configurations of the NVT.

4. Verify the GetDNSResponse from NVT (DNSInformation[FromDHCP = true or false, SearchDomain = domain to search if hostname is not fully qualified, DNSFromDHCP = list of DNS Servers obtained from DHCP, DNSManual = list of DNS Servers manually configured]).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetDNSResponse message.

The DUT did not send correct information (i.e. DNSInformation[FromDHCP = true or false, SearchDomain = domain to search if hostname is not fully qualified, DNSFromDHCP = list of DNS Servers obtained from DHCP, DNSManual = list of DNS Servers manually configured]) in the GetDNSResponse message.

Note: See Annex A.19 for valid expression in terms of empty IP address.

6.2.5 NVT SET DNS CONFIGURATION - SEARCHDOMAIN

Test Label: Device Management NVT Network Command SetDNS SearchDomain Test.

ONVIF Core Specification Coverage: 8.2.4 Set DNS settings

Device Type: NVT

Command Under Test: SetDNS

WSDL Reference: devicemgmt.wsdl

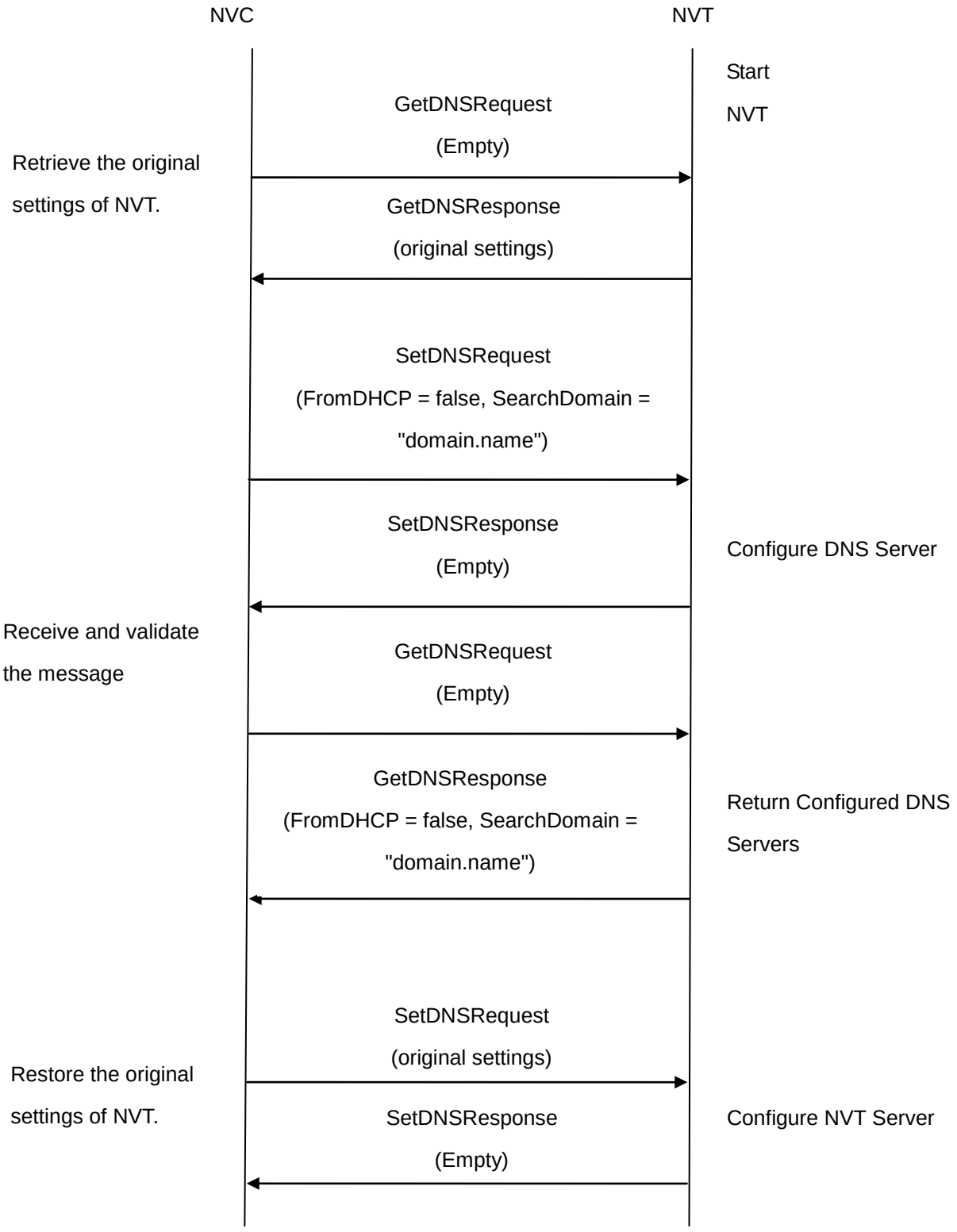
Requirement Level: MUST

Test Purpose: To configure DNS Search Domain setting in NVT through SetDNS command.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start
NVT
GetDNSRequest
(Empty)
Retrieve the original
settings of NVT.
GetDNSResponse
(original settings)
SetDNSRequest
(FromDHCP = false, SearchDomain =
"domain.name")
SetDNSResponse
(Empty)
Configure DNS Server
Receive and validate
the message
GetDNSRequest
(Empty)
GetDNSResponse
(FromDHCP = false, SearchDomain =
"domain.name")
Return Configured DNS
Servers
SetDNSRequest
(original settings)
Restore the original
settings of NVT.
SetDNSResponse
(Empty)
Configure NVT Server

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetDNSRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetDNSRequest message (FromDHCP = false, SearchDomain = "domain.name").
5. Verify that the NVT sends SetDNSResponse (empty message)
6. Verify the DNS configurations in NVT through GetDNSRequest.
7. NVT sends its DNS configurations in the GetDNSResponse message (DNSInformation[FromDHCP = false, SearchDomain = "domain.name"]).
8. NVC will invoke SetDNSRequest message to restore the original settings of NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetDNSResponse message.

The DUT did not send GetDNSResponse message.

The DUT did not send correct information (i.e. DNSInformation[FromDHCP = false, SearchDomain = "domain.name"]) in the GetDNSResponse message.

6.2.6 NVT SET DNS CONFIGURATION - DNSMANUAL IPV4

Test Label: Device Management NVT Network Command SetDNS Test.(DNSManual = IPv4 address)

ONVIF Core Specification Coverage: 8.2.4 Set DNS settings

Device Type: NVT

Command Under Test: SetDNS

WSDL Reference: devicemgmt.wsdl

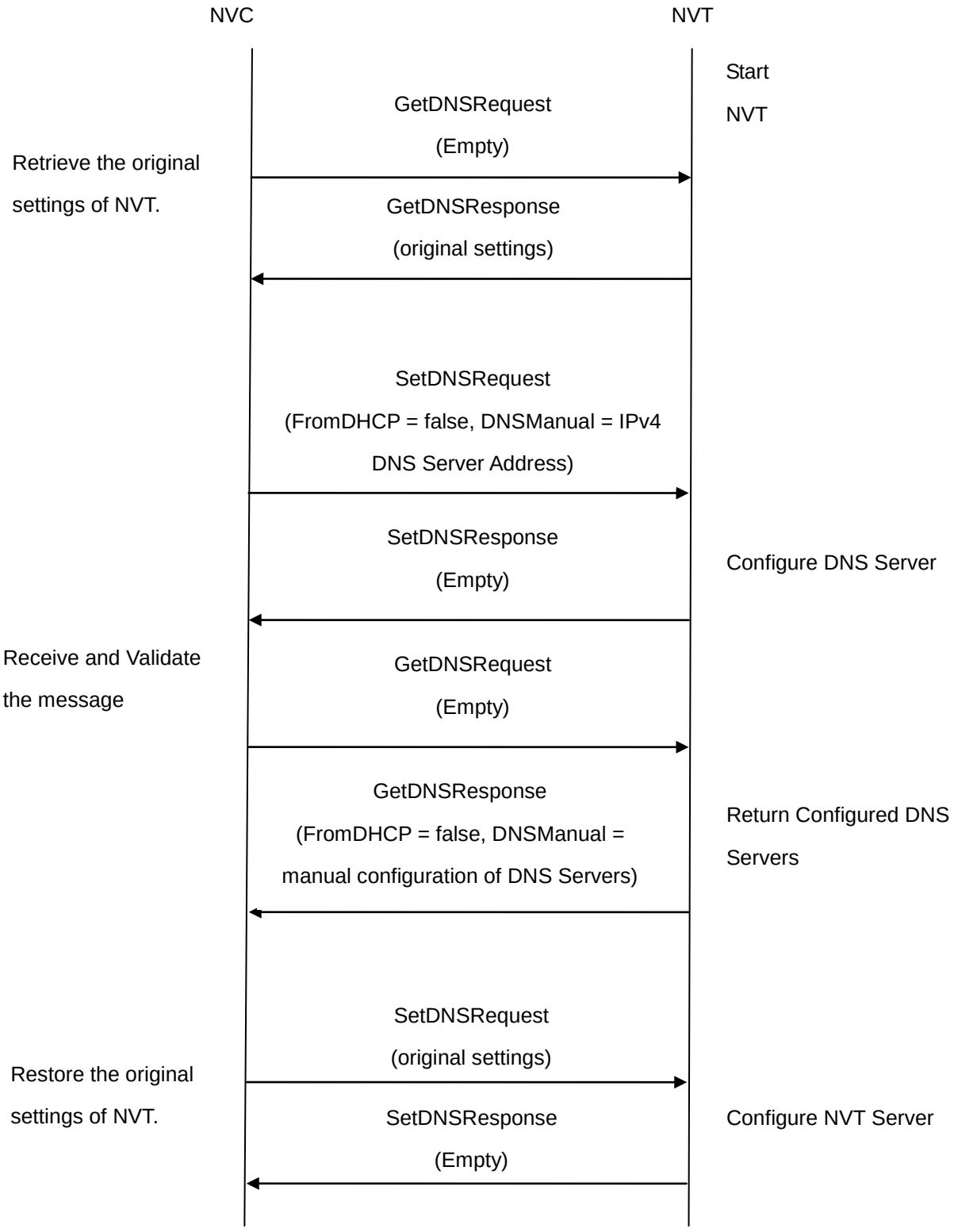
Requirement Level: MUST

Test Purpose: To configure IPv4 DNS server address setting in NVT through SetDNS command.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start
NVT
GetDNSRequest
(Empty)
Retrieve the original
settings of NVT.
GetDNSResponse
(original settings)
SetDNSRequest
(FromDHCP = false, DNSManual = IPv4
DNS Server Address)
SetDNSResponse
(Empty)
Configure DNS Server
Receive and Validate
the message
GetDNSRequest
(Empty)
GetDNSResponse
(FromDHCP = false, DNSManual =
manual configuration of DNS Servers)
Return Configured DNS
Servers
SetDNSRequest
(original settings)
Restore the original
settings of NVT.
SetDNSResponse
(Empty)
Configure NVT Server

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetDNSRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetDNSRequest message (FromDHCP = false, DNSManual = "IPv4", "10.1.1.1").
5. Verify that the NVT sends SetDNSResponse (empty message)
6. Verify the DNS configurations in NVT through GetDNSRequest.
7. NVT sends its DNS configurations in the GetDNSResponse message (DNSInformation[FromDHCP = false, DNSManual = "IPv4", "10.1.1.1"]).
8. NVC will invoke SetDNSRequest message to restore the original settings of NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetDNSResponse message.

The DUT did not send GetDNSResponse message.

The DUT did not send correct information (i.e. DNSInformation[FromDHCP = false, DNSManual = "IPv4", "10.1.1.1"]) in the GetDNSResponse message.

Note: See Annex A.9 for Valid IPv4 Address definition.

See Annex A.19 for valid expression in terms of empty IP address.

6.2.7 NVT SET DNS CONFIGURATION - DNSMANUAL IPV6

Test Label: Device Management NVT Network Command SetDNS Test. (DNSManual = IPv6 address)

ONVIF Core Specification Coverage: 8.2.4 Set DNS settings

Device Type: NVT

Command Under Test: SetDNS

WSDL Reference: devicemgmt.wsdl

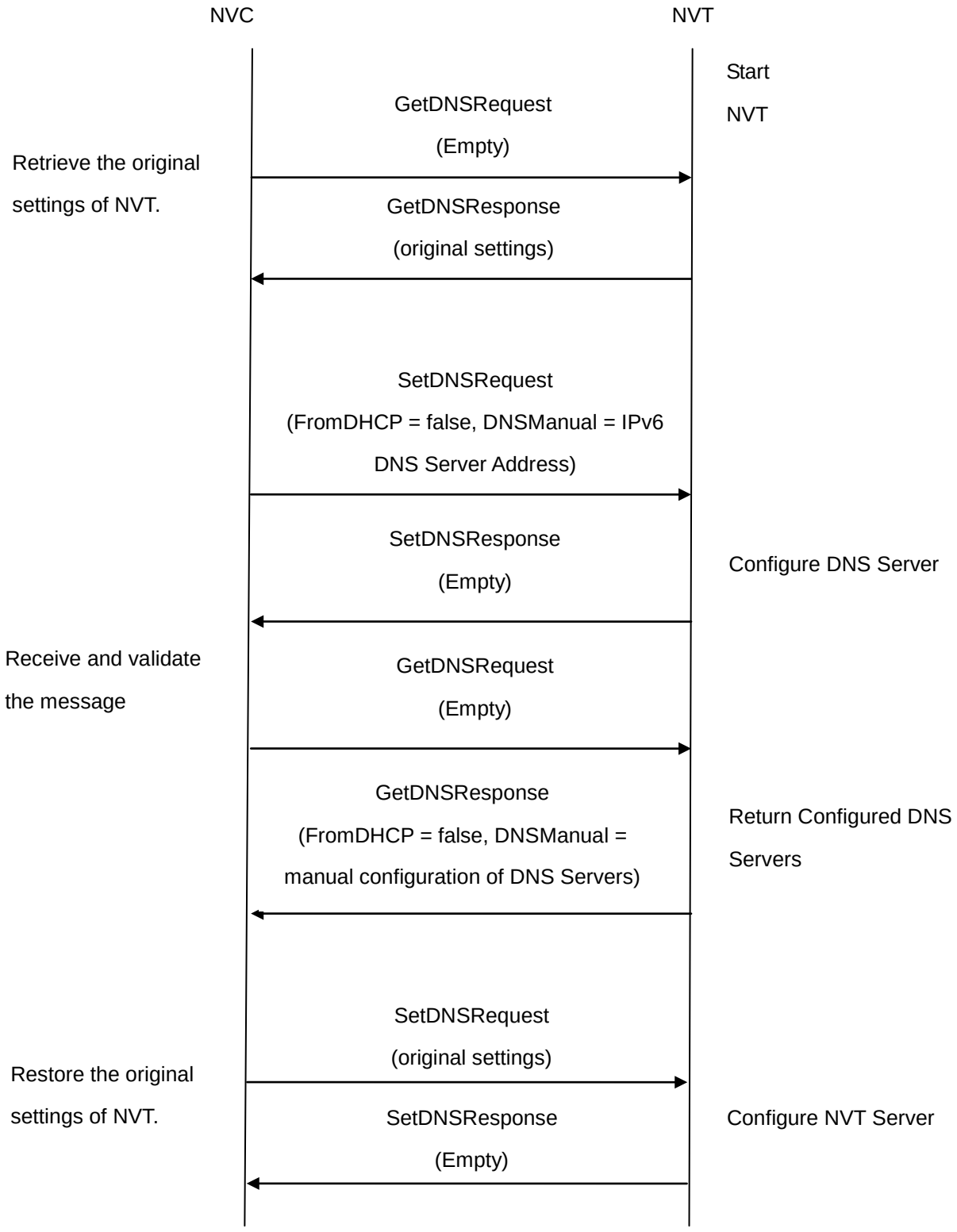
Requirement Level: MUST IF IMPLEMENTED (IPv6)

Test Purpose: To configure IPv6 DNS server address setting in NVT through SetDNS command.

Pre-Requisite: IPv6 is implemented by NVT.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start
NVT
GetDNSRequest
(Empty)
Retrieve the original
settings of NVT.
GetDNSResponse
(original settings)
SetDNSRequest
(FromDHCP = false, DNSManual = IPv6
DNS Server Address)
SetDNSResponse
(Empty)
Configure DNS Server
Receive and validate
the message
GetDNSRequest
(Empty)
GetDNSResponse
(FromDHCP = false, DNSManual =
manual configuration of DNS Servers)
Return Configured DNS
Servers
SetDNSRequest
(original settings)
Restore the original
settings of NVT.
SetDNSResponse
(Empty)
Configure NVT Server

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetDNSRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetDNSRequest message (FromDHCP = false, DNSManual = "IPv6", "2001:1:1:1:1:1:1:1").
5. Verify that the NVT sends SetDNSResponse (empty message)
6. Verify the DNS configurations in NVT through GetDNSRequest.
7. NVT sends its DNS configurations in the GetDNSResponse message (DNSInformation[FromDHCP = false, DNSManual = "IPv6", "2001:1:1:1:1:1:1:1"]).
8. NVC will invoke SetDNSRequest message to restore the original settings of NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetDNSResponse message.

The DUT did not send GetDNSResponse message.

The DUT did not send correct information (i.e. DNSInformation[FromDHCP = false, DNSManual = "IPv6", "2001:1:1:1:1:1:1:1"]) in the GetDNSResponse message.

6.2.8 NVT SET DNS CONFIGURATION - FROMDHCP

Test Label: Device Management NVT Network Command SetDNS FromDHCP Test.

ONVIF Core Specification Coverage: 8.2.4 Set DNS settings

Device Type: NVT

Command Under Test: SetDNS

WSDL Reference: devicemgmt.wsdl

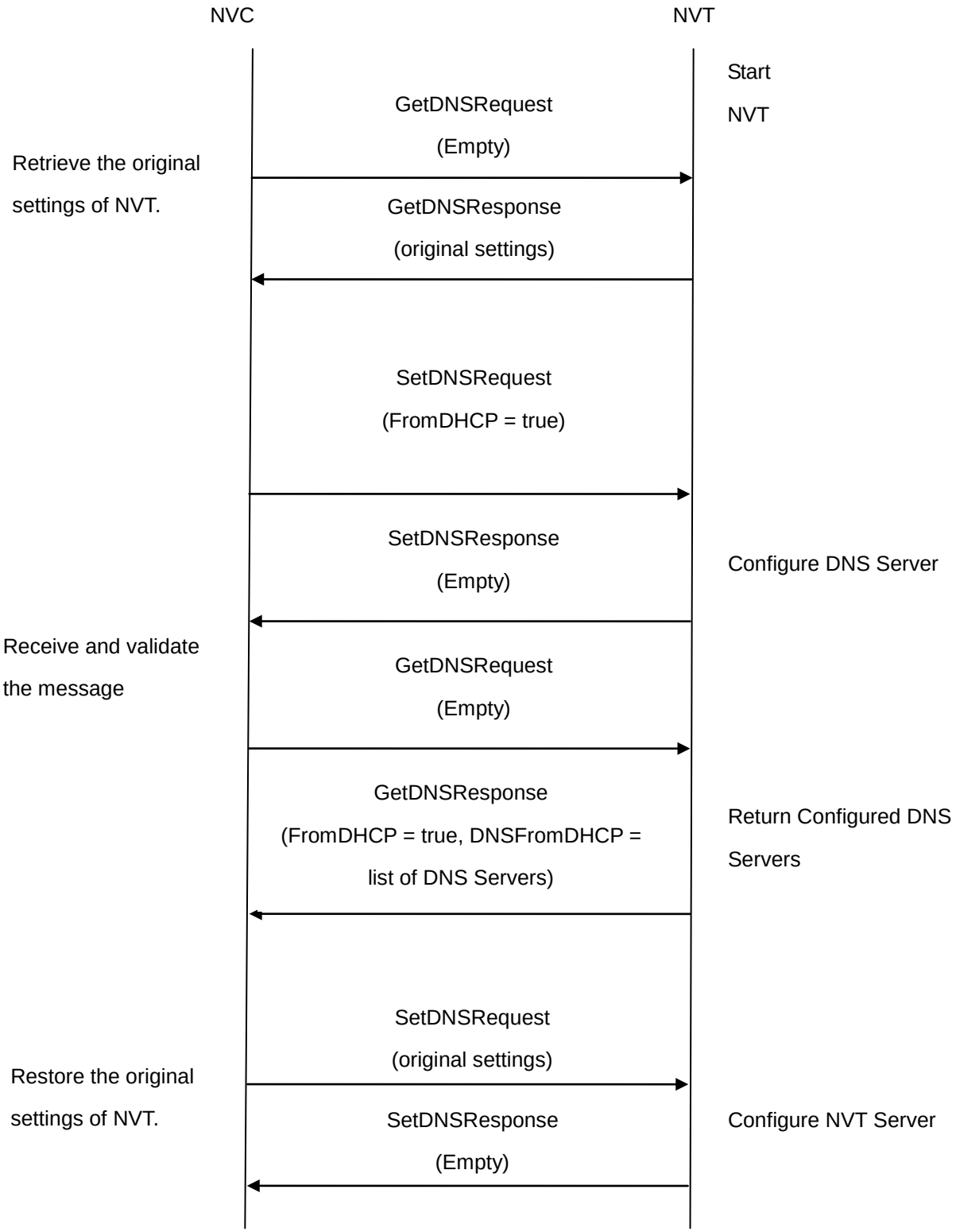
Requirement Level: MUST

Test Purpose: To configure DNS FromDHCP setting in NVT through SetDNS command.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start
NVT
GetDNSRequest
(Empty)
Retrieve the original settings of NVT.
GetDNSResponse
(original settings)
SetDNSRequest
(FromDHCP = true)
SetDNSResponse
(Empty)
Configure DNS Server
Receive and validate the message
GetDNSRequest
(Empty)
GetDNSResponse
(FromDHCP = true, DNSFromDHCP = list of DNS Servers)
Return Configured DNS Servers
SetDNSRequest
(original settings)
Restore the original settings of NVT.
SetDNSResponse
(Empty)
Configure NVT Server

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetDNSRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetDNSRequest message (FromDHCP = true).
5. Verify that the NVT sends SetDNSResponse (empty message)
6. Verify the DNS configurations in NVT through GetDNSRequest.
7. NVT sends its DNS configurations in the GetDNSResponse message (DNSInformation[FromDHCP = true, DNSFromDHCP = list of DNS Servers obtained from DHCP]).
8. NVC will invoke SetDNSRequest message to restore the original settings of NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetDNSResponse message.

The DUT did not send GetDNSResponse message.

The DUT did not send correct information (i.e. DNSInformation[FromDHCP = true, DNSFromDHCP = list of DNS Servers obtained from DHCP]) in the GetDNSResponse message.

6.2.9 NVT SET DNS CONFIGURATION - DNSMANUAL INVALID IPV4

Test Label: Device Management NVT Network Command SetDNS Test.(DNSManual = invalid IPv4 address)

ONVIF Core Specification Coverage: 8.2.4 Set DNS settings

Device Type: NVT

Command Under Test: SetDNS

WSDL Reference: devicemgmt.wsdl

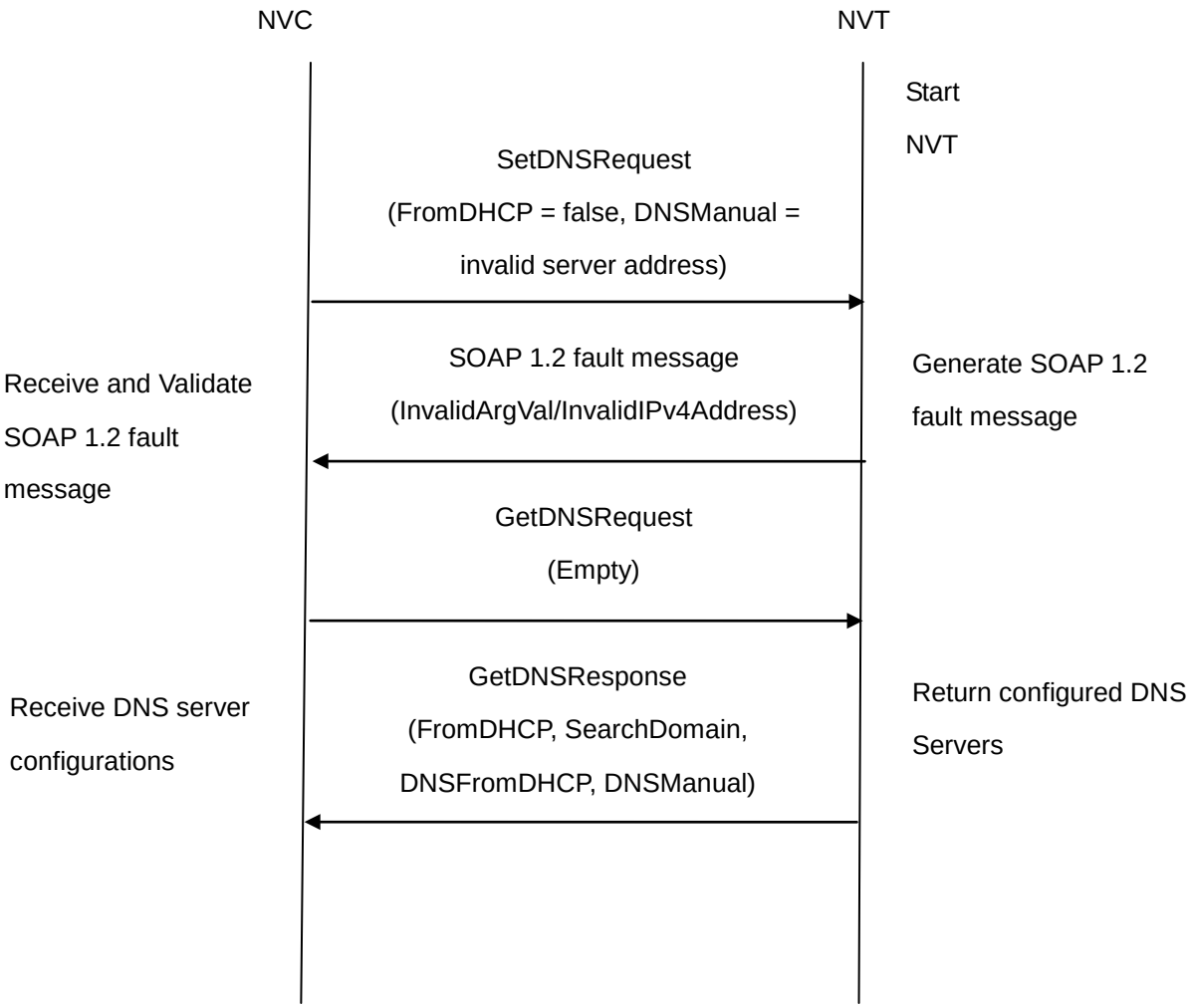
Requirement Level: SHOULD

Test Purpose: To verify behaviour of NVT for invalid DNS IPv4 address configuration.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetDNSRequest message (FromDHCP = false, DNSManual = “IPv4”, “10.1.1”).
4. Verify that the NVT generates SOAP 1.2 fault message (InvalidArgVal/InvalidIPv4Address).
5. Retrieve DNS configurations from NVT through GetDNSRequest.
6. NVT sends valid DNS configurations in the GetDNSResponse message (DNSInformation[FromDHCP=true or false, SearchDomain = domain to search if hostname is not fully qualified, DNSFromDHCP = list of DNS Servers obtained from DHCP, DNSManual = list of manual DNS Servers]).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct fault code in the SOAP fault message (InvalidArgVal/InvalidIPv4Address).

The DUT did not GetDNSResponse message.

The DUT returned “10.1.1” as DNS Server address.

The DUT did not send correct information (i.e. DNSInformation[FromDHCP=true or false, SearchDomain = domain to search if hostname is not fully qualified, DNSFromDHCP = list of DNS Servers obtained from DHCP, DNSManual = list of manual DNS Servers]) in the GetDNSResponse message.

Note: See Annex A.9 for Invalid IPv4 Address and SOAP 1.2 fault message definitions.

See Annex A.19 for valid expression in terms of empty IP address.

6.2.10 NVT SET DNS CONFIGURATION - DNSMANUAL INVALID IPV6

Test Label: Device Management NVT Network Command SetDNS Test. (DNSManual = invalid IPv6 address)

ONVIF Core Specification Coverage: 8.2.4 Set DNS settings

Device Type: NVT

Command Under Test: SetDNS

WSDL Reference: devicemgmt.wsdl

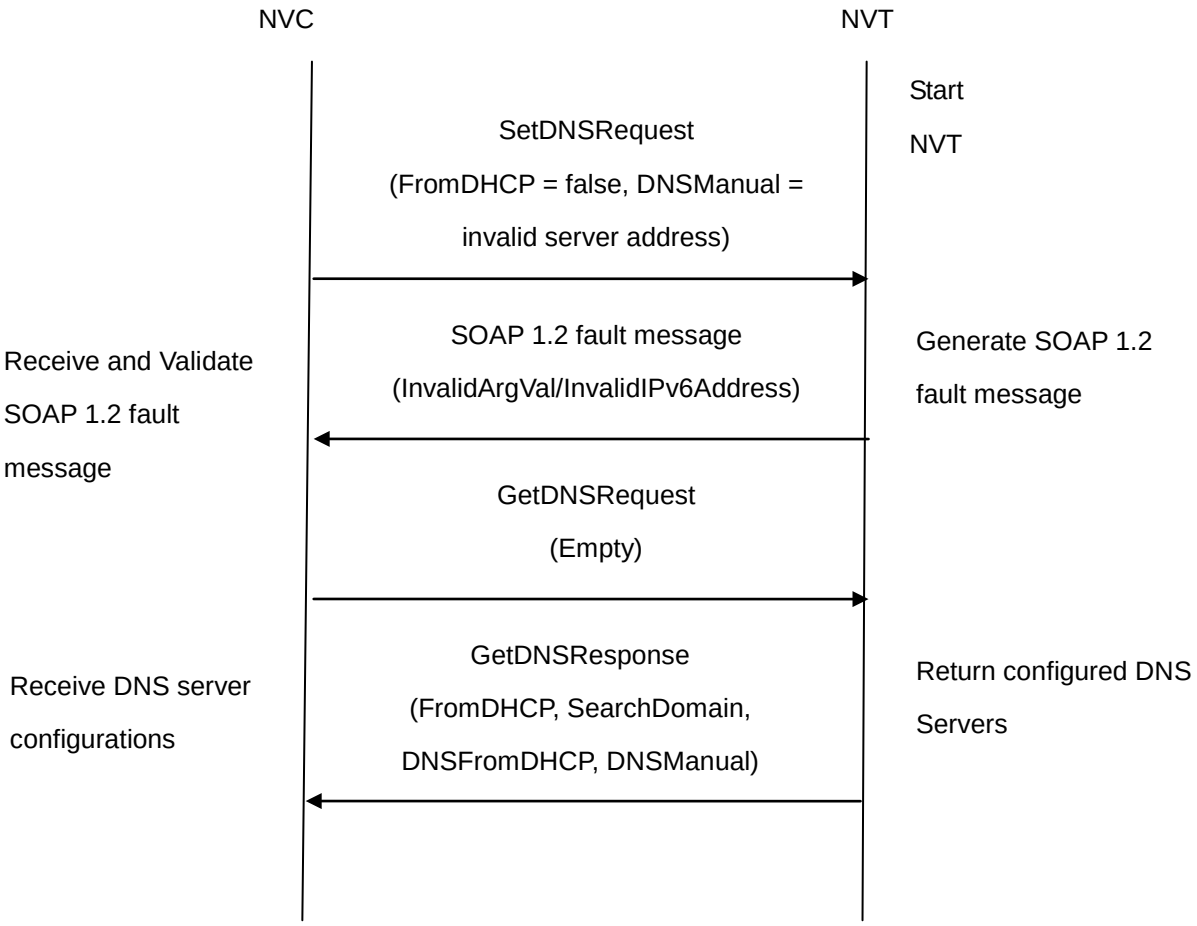
Requirement Level: SHOULD IF IMPLEMENTED (IPv6)

Test Purpose: To verify behaviour of NVT for invalid DNS IPv6 address configuration.

Pre-Requisite: IPv6 is implemented by NVT.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetDNSRequest message (FromDHCP = false, DNSManual = “IPv6”, “FF02:1”).
4. Verify that the NVT generates SOAP 1.2 fault message (InvalidArgVal/InvalidIPv6Address).
5. Retrieve DNS configurations from NVT through GetDNSRequest.
6. NVT sends valid DNS configurations in the GetDNSResponse message (DNSInformation[FromDHCP=true or false, SearchDomain = domain to search if hostname is not fully qualified, DNSFromDHCP = list of DNS Servers obtained from DHCP, DNSManual = list of manual DNS Servers]).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct fault code in the SOAP fault message (InvalidArgVal/InvalidIPv6Address).

The DUT did not GetDNSResponse message.

The DUT returned “FF02:1” as DNS Server address.

The DUT did not send correct information (i.e. DNSInformation[FromDHCP=true or false, SearchDomain = domain to search if hostname is not fully qualified, DNSFromDHCP = list of DNS Servers obtained from DHCP, DNSManual = list of manual DNS Servers]) in the GetDNSResponse message.

6.2.11 NVT GET NTP CONFIGURATION

Test Label: Device Management NVT Network Command GetNTP Test

ONVIF Core Specification Coverage: 8.2.5 Get NTP settings

Device Type: NVT

Command Under Test: GetNTP

WSDL Reference: devicemgmt.wsdl

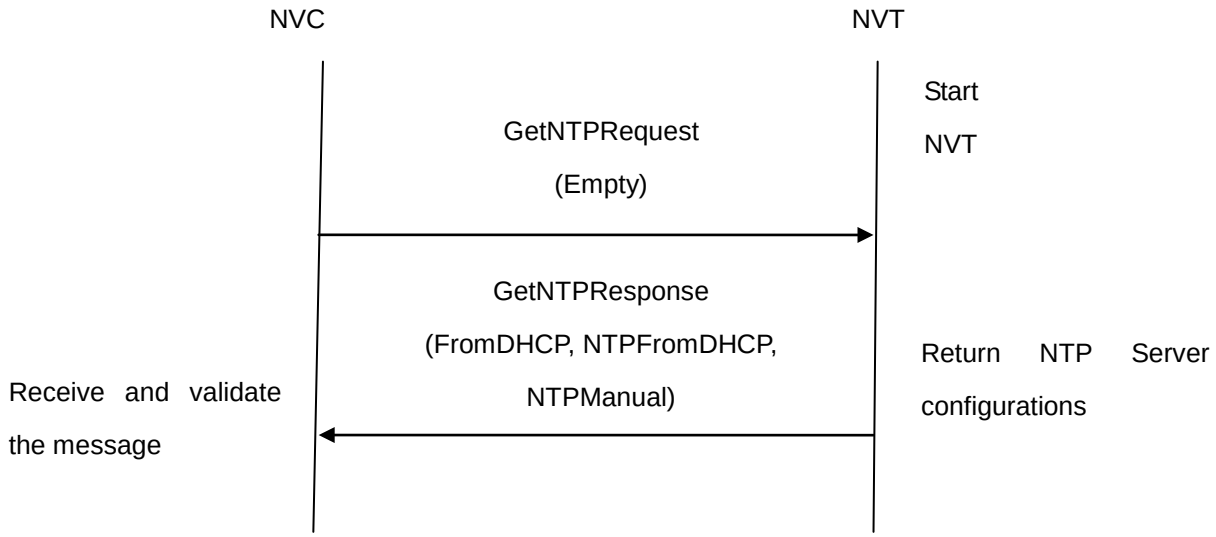
Requirement Level: MUST IF SUPPORTED (NTP)

Test Purpose: To retrieve NTP Server settings of the NVT through GetNTP command.

Pre-Requisite: NTP is supported by NVT.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNTPRequest message to retrieve NTP Server settings of the NVT.
4. Verify the GetNTPResponse from NVT (NTPInformation[FromDHCP = true or false, NTPFromDHCP = list of NTP Servers obtained from DHCP, NTPManual = list of NTP Servers manually configured]).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetNTPResponse message.

The DUT did not send correct information (i.e. NTPInformation[FromDHCP = true or false, NTPFromDHCP = list of NTP Servers obtained from DHCP, NTPManual = list of NTP Servers manually configured]) in the GetNTPResponse message.

Note: See Annex A.19 for valid expression in terms of empty IP address

6.2.12 NVT SET NTP CONFIGURATION - NTPMANUAL IPV4

Test Label: Device Management NVT Network Command SetNTP Test. (NTPManual = IPv4 address)

ONVIF Core Specification Coverage: 8.2.6 Set NTP settings

Device Type: NVT

Command Under Test: SetNTP

WSDL Reference: devicemgmt.wsdl

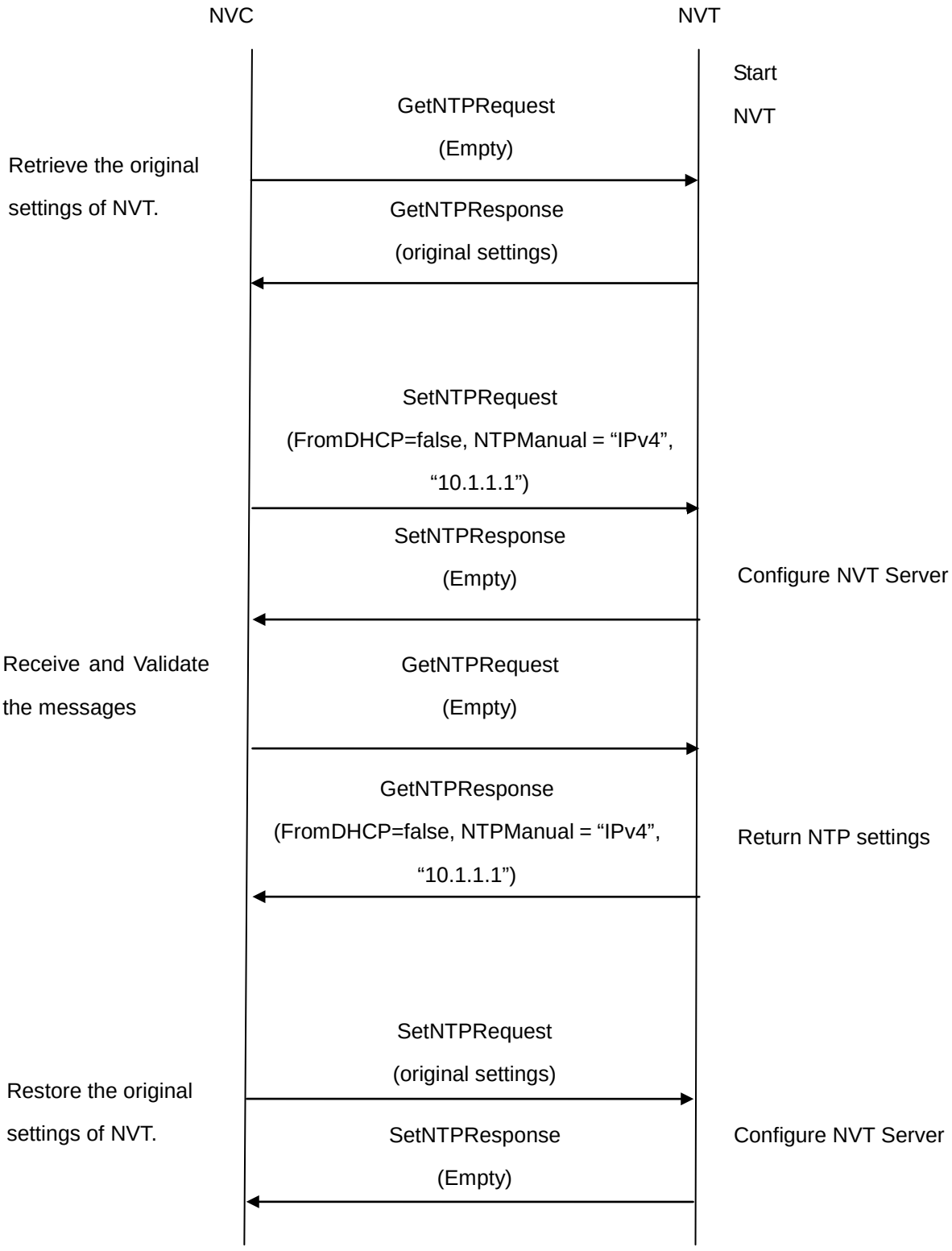
Requirement Level: MUST IF SUPPORTED (NTP)

Test Purpose: To configure NTP IPv4 address settings on an NVT through SetNTP command.

Pre-Requisite: NTP is supported by NVT.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.

2. Start an NVT.
3. NVC will invoke GetNTPRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetNTPRequest message (FromDHCP = false, NTPManual [Type = “IPv4”, IPv4Address = “10.1.1.1”]).
5. Verify that the NVT sends SetNTPResponse (empty message).
6. Verify the NTP Server settings in NVT through GetNTPRequest message.
7. NVT sends its NTP Server settings in the GetNTPResponse message (NTPInformation[FromDHCP= false, NTPManual [Type = “IPv4”, IPv4Address = “10.1.1.1”]]).
8. NVC will invoke SetNTPRequest message to restore the original settings of NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNTPResponse message in step-5.

The DUT did not send GetNTPResponse message in step-7.

The DUT did not send correct NTP Server information (i.e. NTPInformation[FromDHCP=false, NTPManual [Type = “IPv4”, IPv4Address = “10.1.1.1”]]) in GetNTPResponse message in step-6.

Note: See Annex A.9 for Valid IPv4 Address definition.

See Annex A.19 for valid expression in terms of empty IP address.

6.2.13 NVT SET NTP CONFIGURATION - NTPMANUAL IPV6

Test Label: Device Management NVT Network Command SetNTP Test.(NTPManual = IPv6 address)

ONVIF Core Specification Coverage: 8.2.6 Set NTP settings

Device Type: NVT

Command Under Test: SetNTP

WSDL Reference: devicemgmt.wsdl

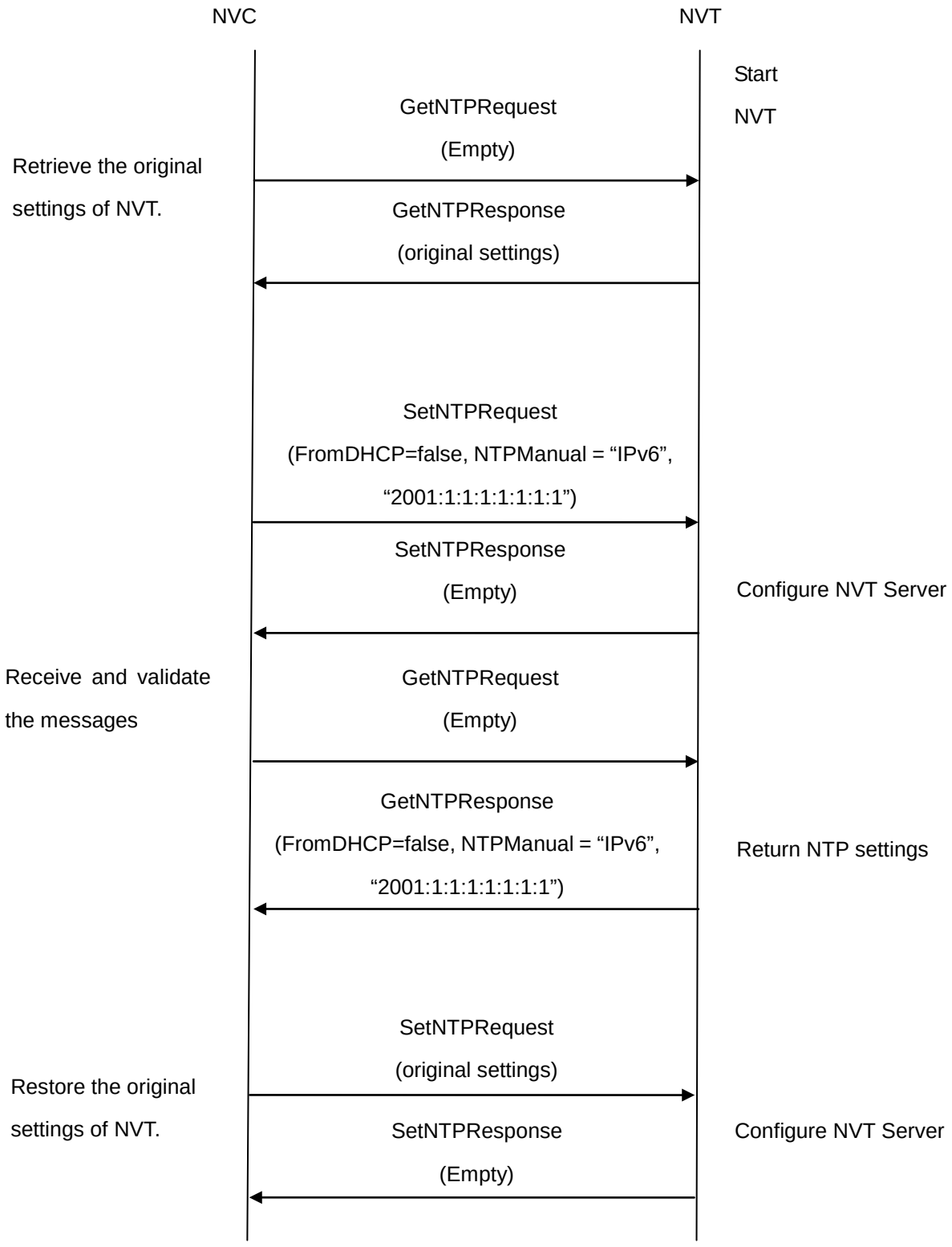
Requirement Level: MUST IF SUPPORTED (NTP) & IMPLEMENTED (IPv6)

Test Purpose: To configure NTP IPv6 address settings on an NVT through SetNTP command.

Pre-Requisite: NTP is supported by NVT. IPv6 is implemented by NVT.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNTPRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetNTPRequest message (FromDHCP = false, NTPManual [Type = “IPv6”, IPv6Address = “2001:1:1:1:1:1:1:1”]).
5. Verify that the NVT sends SetNTPResponse (empty message).
6. Verify the NTP Server settings in NVT through GetNTPRequest message.
7. NVT sends its NTP Server settings in the GetNTPResponse message (NTPInformation[FromDHCP= false, NTPManual [Type = “IPv6”, IPv6Address = “2001:1:1:1:1:1:1:1”]]).
8. NVC will invoke SetNTPRequest message to restore the original settings of NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNTPResponse message in step-5.

The DUT did not send GetNTPResponse message in step-7.

The DUT did not send correct NTP Server information (i.e. NTPInformation[FromDHCP=false, NTPManual [Type = “IPv6”, IPv6Address = “2001:1:1:1:1:1:1:1”]]) in GetNTPResponse message in step-7.

6.2.14 NVT SET NTP CONFIGURATION - FROMDHCP

Test Label: Device Management NVT Network Command SetNTP FromDHCP Test.

ONVIF Core Specification Coverage: 8.2.6 Set NTP settings

Device Type: NVT

Command Under Test: SetNTP

WSDL Reference: devicemgmt.wsdl

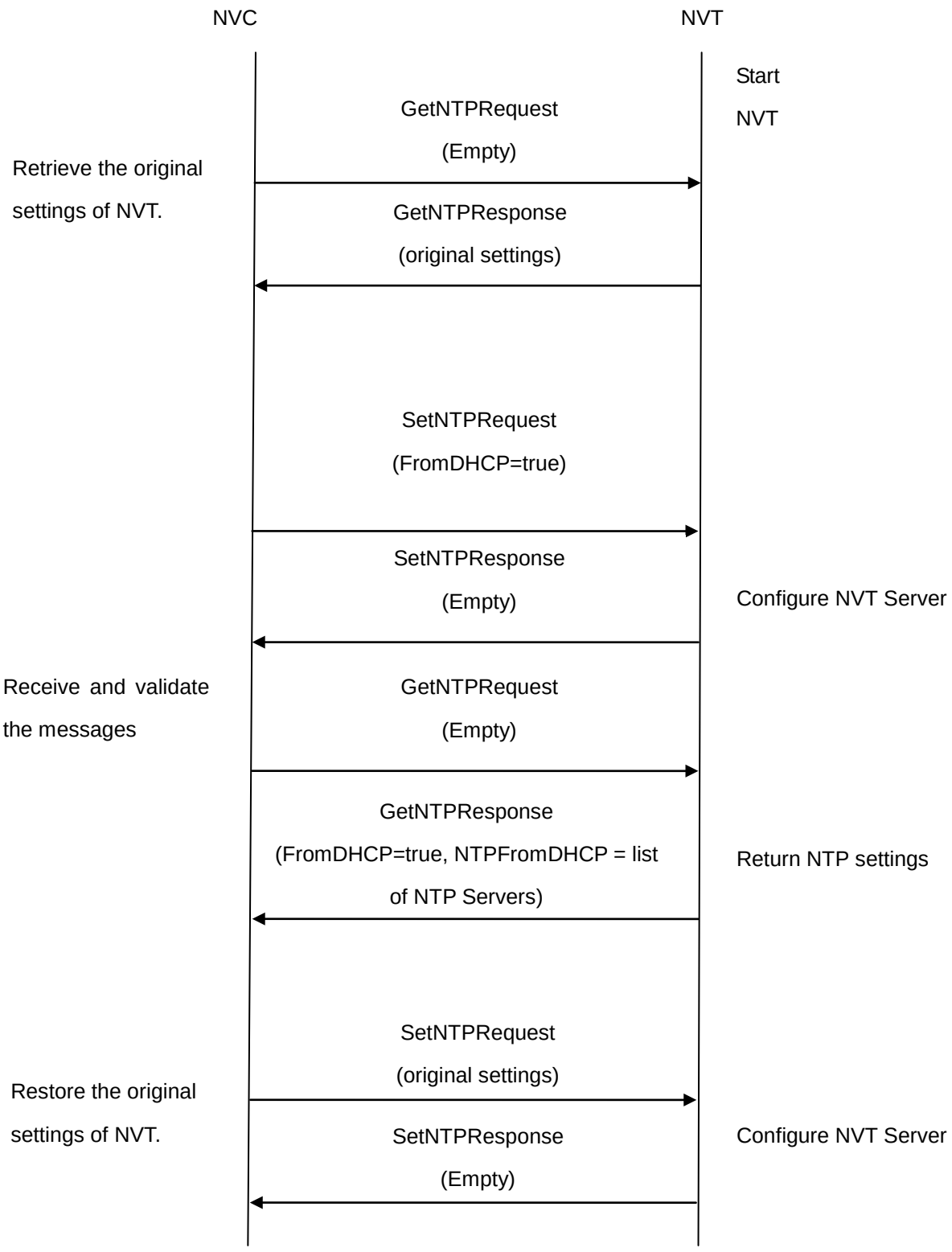
Requirement Level: MUST IF SUPPORTED (NTP)

Test Purpose: To configure NVT's NTP settings via DHCP server using SetNTP command.

Pre-Requisite: NTP is supported by NVT.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.

2. Start an NVT.
3. NVC will invoke GetNTPRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetNTPRequest message (FromDHCP = true).
5. Verify that the NVT sends SetNTPResponse (empty message).
6. Verify the NTP Server settings in NVT through GetNTPRequest message.
7. NVT sends its NTP Server settings in the GetNTPResponse message (NTPInformation[FromDHCP= true, NTPFromDHCP = list of NTP Servers obtained from DHCP]).
8. NVC will invoke SetNTPRequest message to restore the original settings of NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNTPResponse message in step-5.

The DUT did not send GetNTPResponse message in step-7.

The DUT did not send correct NTP Server information (i.e. NTPInformation[FromDHCP=true, NTPFromDHCP = list of NTP Servers obtained from DHCP]) in GetNTPResponse message in step-7.

6.2.15 NVT SET NTP CONFIGURATION - NTPMANUAL INVALID IPV4

Test Label: Device Management NVT Network Command SetNTP Test.(NTPManual = invalid IPv4 address)

ONVIF Core Specification Coverage: 8.2.6 Set NTP settings

Device Type: NVT

Command Under Test: SetNTP

WSDL Reference: devicemgmt.wsdl

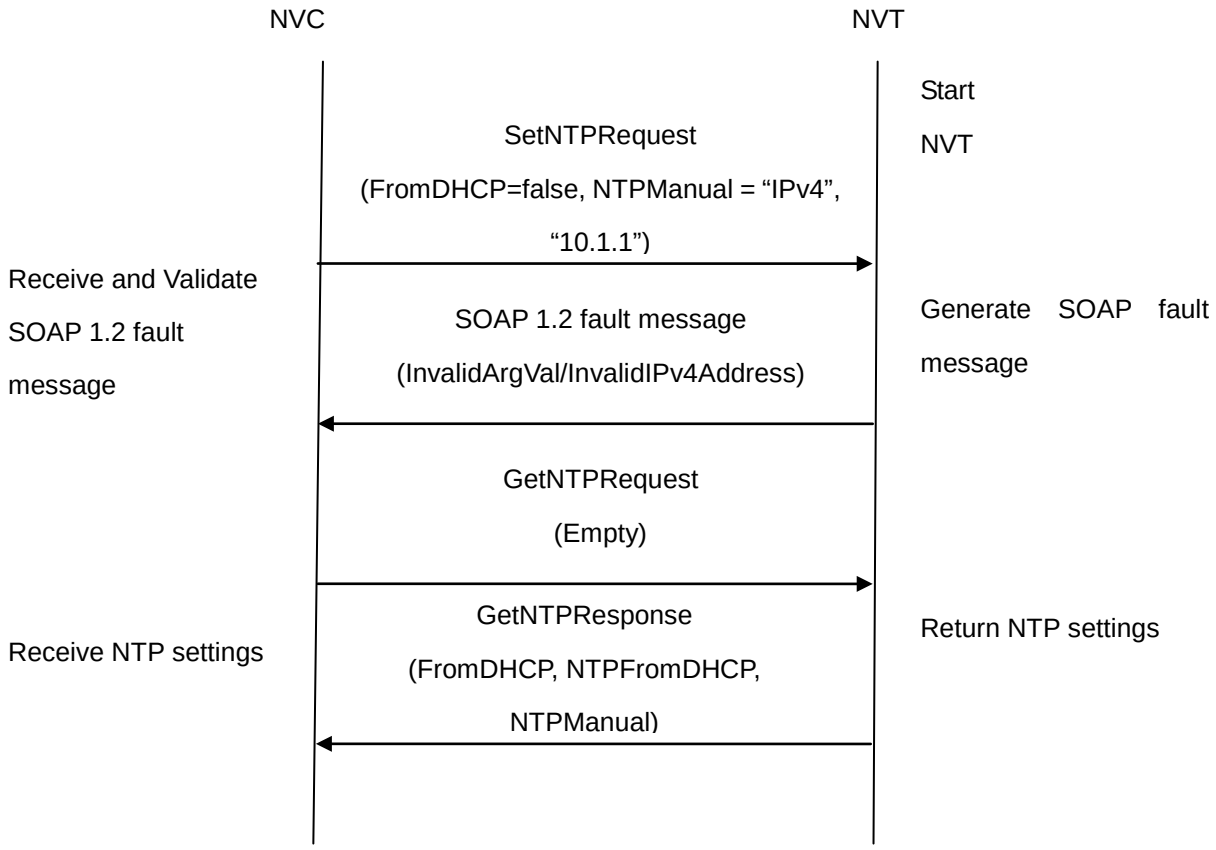
Requirement Level: SHOULD IF SUPPORTED (NTP)

Test Purpose: To verify behaviour of NVT for invalid NTP IPv4 address configuration.

Pre-Requisite: NTP is supported by NVT.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetNTPRequest message (FromDHCP = false, NTPManual [Type = “IPv4”, IPv4Address = “10.1.1”]).
4. Verify that the NVT generates SOAP 1.2 fault message (InvalidArgVal/InvalidIPv4Address).
5. Retrieve NTP Server configurations from NVT through GetNTPRequest message.
6. NVT sends valid NTP Server configurations in the GetNTPResponse message (NTPInformation[FromDHCP = true or false, NTPFromDHCP = list of NTP Servers obtained from DHCP, NTPManual = list of NTP Servers manually configured]).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct fault code in the SOAP fault message (InvalidArgVal/InvalidIPv4Address).

The DUT did not GetNTPResponse message.

The DUT returned “10.1.1” as NTP Server address.

The DUT did not send correct NTP Server information (i.e. NTPInformation[FromDHCP = true or false, NTPFromDHCP = list

of NTP Servers obtained from DHCP, NTPManual = list of NTP Servers manually configured]) in GetNTPResponse message.

Note: See Annex A.9 for Invalid IPv4 Address and SOAP 1.2 fault message definitions.

6.2.16 NVT SET NTP CONFIGURATION - NTPMANUAL INVALID IPV6

Test Label: Device Management NVT Network Command SetNTP Test.(NTPManual = invalid IPv6 address)

ONVIF Core Specification Coverage: 8.2.6 Set NTP settings

Device Type: NVT

Command Under Test: SetNTP

WSDL Reference: devicemgmt.wsdl

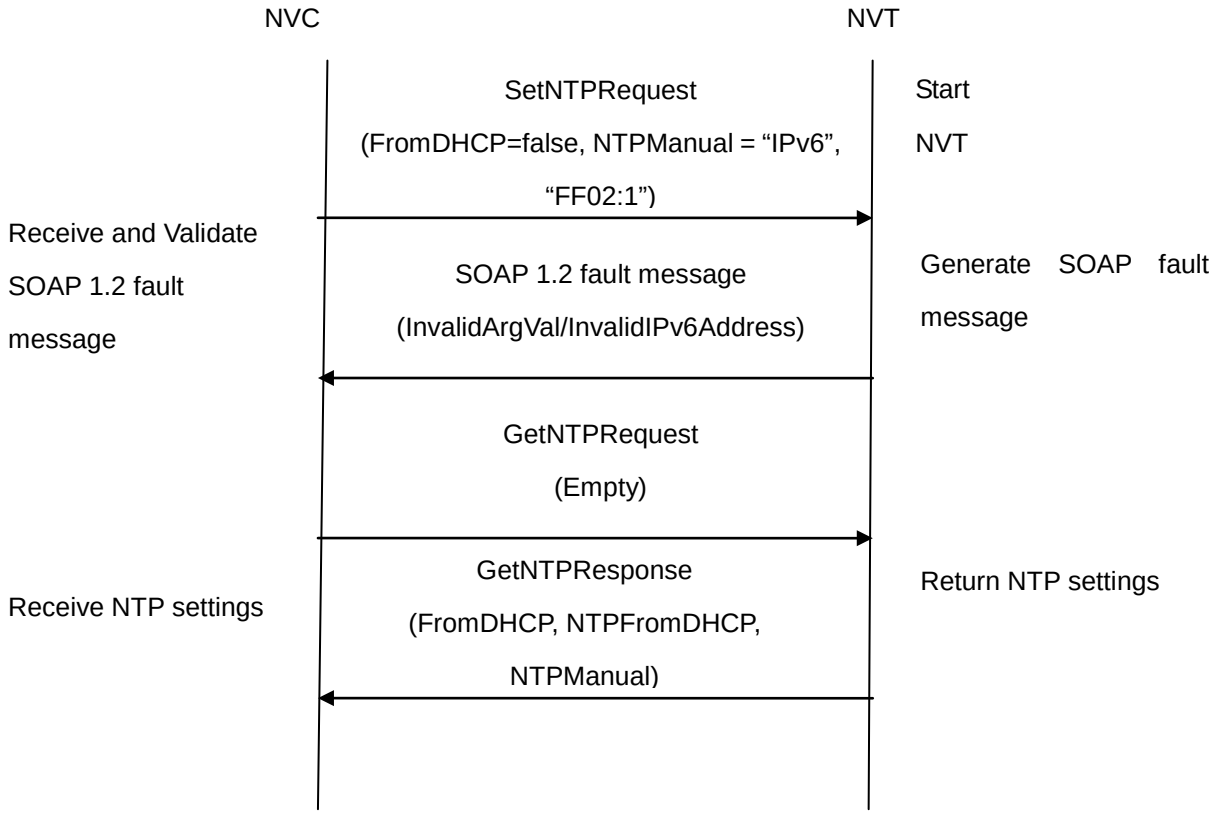
Requirement Level: SHOULD IF SUPPORTED (NTP) & IMPLEMENTED (IPv6)

Test Purpose: To verify behaviour of NVT for invalid NTP IPv6 address configuration.

Pre-Requisite: NTP is supported by NVT. IPv6 is implemented by NVT.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetNTPRequest message (FromDHCP = false, NTPManual [Type = “IPv6”, IPv6Address = “FF02:1”]).
4. Verify that the NVT generates SOAP 1.2 fault message (InvalidArgVal/InvalidIPv6Address).
5. Retrieve NTP Server configurations from NVT through GetNTPRequest message.
6. NVT sends valid NTP Server configurations in the GetNTPResponse message (NTPInformation[FromDHCP = true or false, NTPFromDHCP = list of NTP Servers obtained from DHCP, NTPManual = list of NTP Servers manually configured]).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct fault code in the SOAP fault message (InvalidArgVal/InvalidIPv4Address).

The DUT did not GetNTPResponse message.

The DUT returned “FF02:1” as NTP Server address.

The DUT did not send correct NTP Server information (i.e. NTPInformation[FromDHCP = true or false, NTPFromDHCP = list

of NTP Servers obtained from DHCP, NTPManual = list of NTP Servers manually configured]) in GetNTPResponse message.

6.2.17 NVT GET NETWORK INTERFACE CONFIGURATION

Test Label: Device Management NVT Network Command GetNetworkInterfaces Test.

ONVIF Core Specification Coverage: 8.2.9 Get network interface configuration

Device Type: NVT

Command Under Test: GetNetworkInterfaces

WSDL Reference: devicemgmt.wsdl

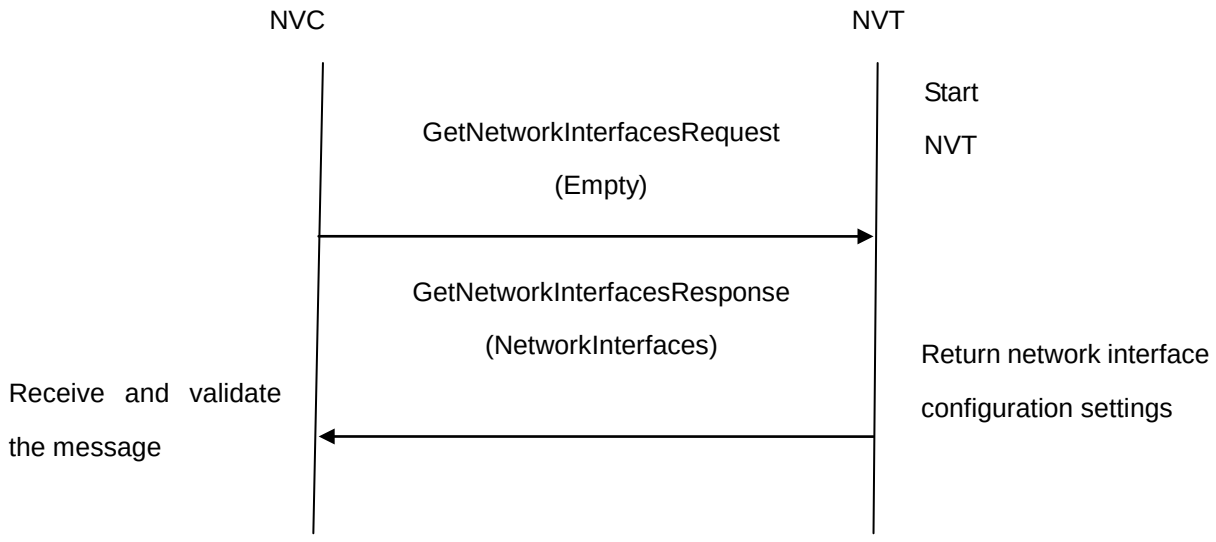
Requirement Level: MUST

Test Purpose: To retrieve network interface configurations of NVT through GetNetworkInterfaces command.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNetworkInterfacesRequest message to retrieve network interface configuration settings of the NVT.
4. Verify the GetNetworkInterfacesResponse from NVT (NetworkInterfaces = list of network interfaces).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetNetworkInterfacesResponse message.

The DUT did not send correct network interface information (i.e. NetworkInterfaces = list of network interfaces) in GetNetworkInterfacesResponse message.

6.2.18 NVT SET NETWORK INTERFACE CONFIGURATION - IPV4

Test Label: Device Management NVT Network Command SetNetworkInterfaces Test. (for IPv4 address)

ONVIF Core Specification Coverage: 8.2.10 Set network interface configuration

Device Type: NVT

Command Under Test: SetNetworkInterfaces

WSDL Reference: devicemgmt.wsdl

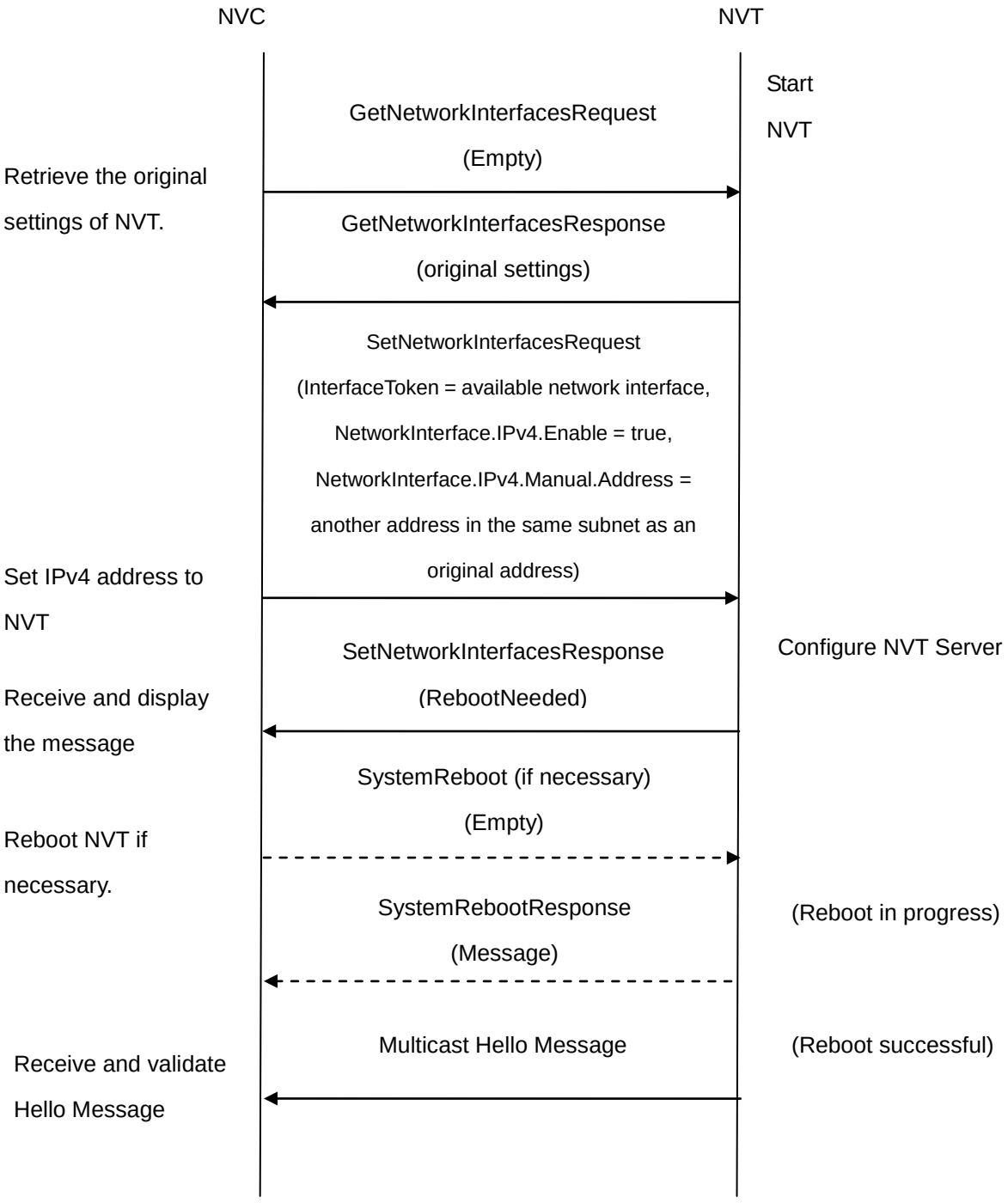
Requirement Level: MUST

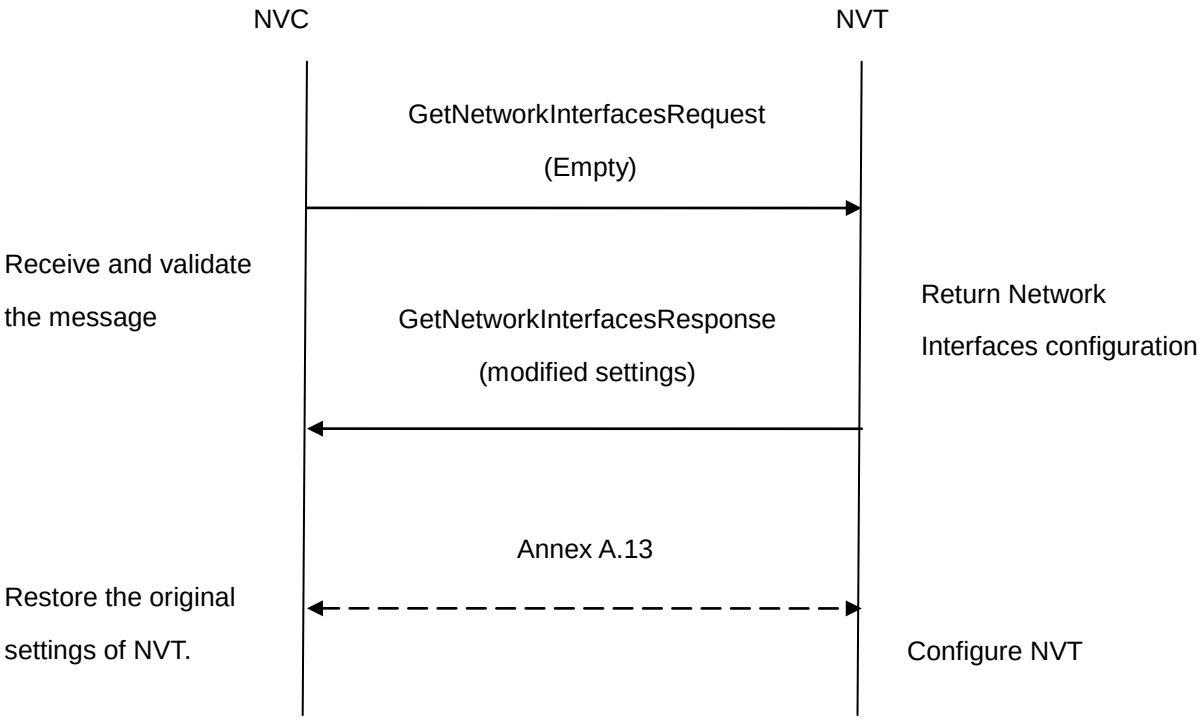
Test Purpose: To configure IPv4 address setting on an NVT through SetNetworkInterfaces command.

Pre-Requisite: NVC knows the available network interface token of NVT.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
GetNetworkInterfacesRequest
(Empty)
Retrieve the original settings of NVT.
GetNetworkInterfacesResponse
(original settings)
SetNetworkInterfacesRequest
(InterfaceToken = available network interface,
NetworkInterface.IPv4.Enable = true,
NetworkInterface.IPv4.Manual.Address =
another address in the same subnet as an
original address)
Set IPv4 address to NVT
SetNetworkInterfacesResponse
(RebootNeeded)
Configure NVT Server
Receive and display the message
SystemReboot (if necessary)
(Empty)
Reboot NVT if necessary.
SystemRebootResponse
(Message)
(Reboot in progress)
Multicast Hello Message
(Reboot successful)
Receive and validate Hello Message



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNetworkInterfacesRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetNetworkInterfacesRequest message to set static IPv4 address to NVT (InterfaceToken = available network interface, NetworkInterface.IPv4.Enable = true, NetworkInterface.IPv4.Manual.Address = another address in the same subnet as an original address).
5. NVT will return SetNetworkInterfacesResponse.
6. If necessary, NVC will invoke SystemReboot message to restart NVT. Otherwise, continue to step-8.
7. NVT will return SystemRebootResponse message.
8. NVT will send Multicast Hello message from newly configured address.
9. NVC will receive and validate Hello message sent from newly configured address by NVT.
10. Verify the network interfaces settings in NVT through GetNetworkInterfacesRequest message.
11. NVT sends its network interfaces settings in the GetNetworkInterfacesResponse message (i.e. NetworkInterface[token = available network interface, IPv4.Enable = true, IPv4.Config.Manual = newly configured address]) from newly configured address.

12. NVC will restore the original settings by following the procedure mentioned in Annex A.13.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNetworkInterfacesResponse message.

The DUT did not send SystemRebootResponse message.

The DUT did not send Multicast Hello message after IP configuration change.

The DUT did not send GetNetworkInterfacesResponse message.

The DUT did not send correct network interface information (i.e. NetworkInterface[token = available network interface, IPv4.Enable = true, IPv4.Config.Manual = newly configured address]) in GetNetworkInterfacesResponse message.

6.2.19 NVT SET NETWORK INTERFACE CONFIGURATION - IPV6

Test Label: Device Management NVT Network Command SetNetworkInterfaces Test.(for IPv6 address)

ONVIF Core Specification Coverage: 8.2.10 Set network interface configuration

Device Type: NVT

Command Under Test: SetNetworkInterfaces

WSDL Reference: devicemgmt.wsdl

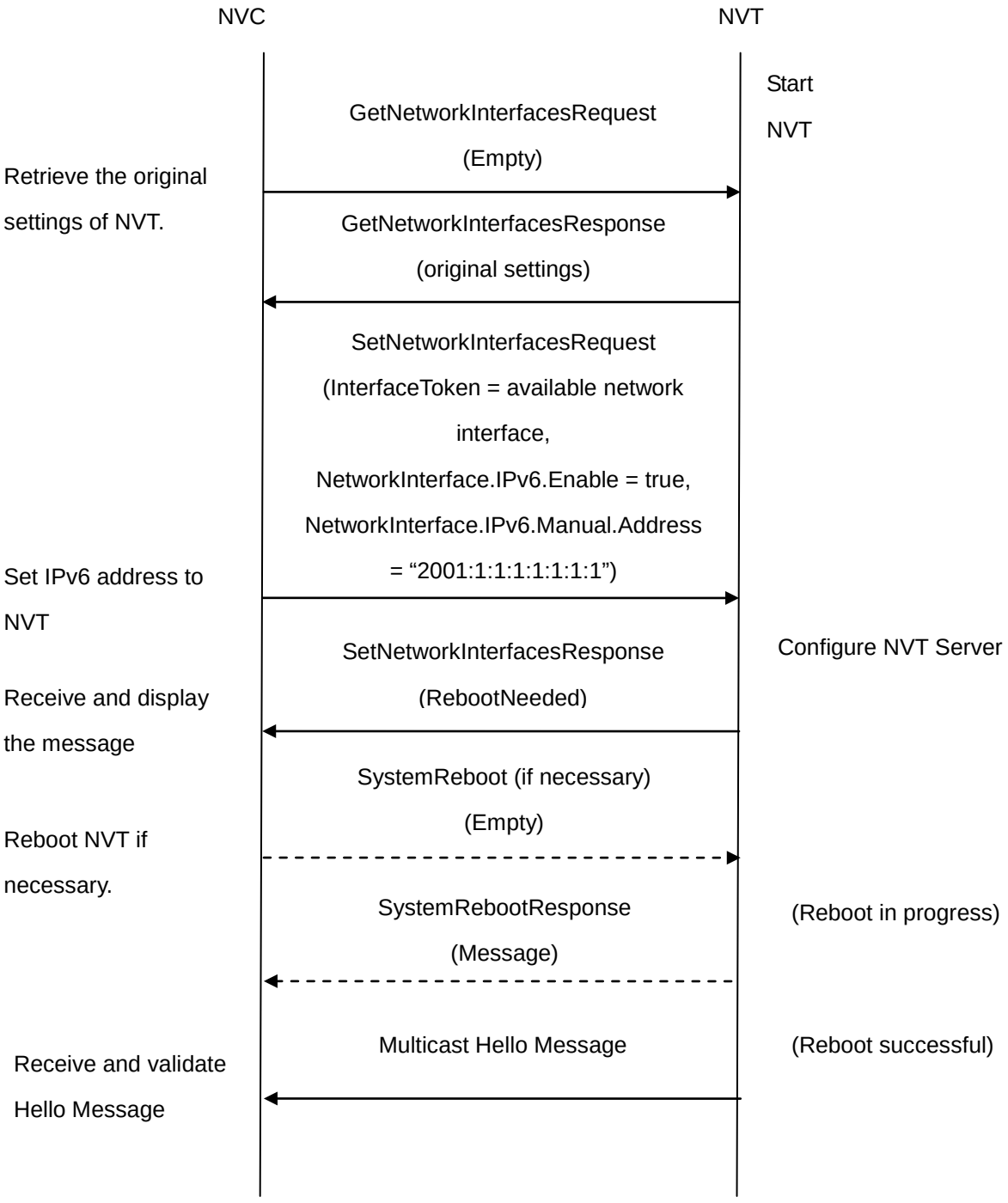
Requirement Level: MUST IF IMPLEMENTED (IPv6)

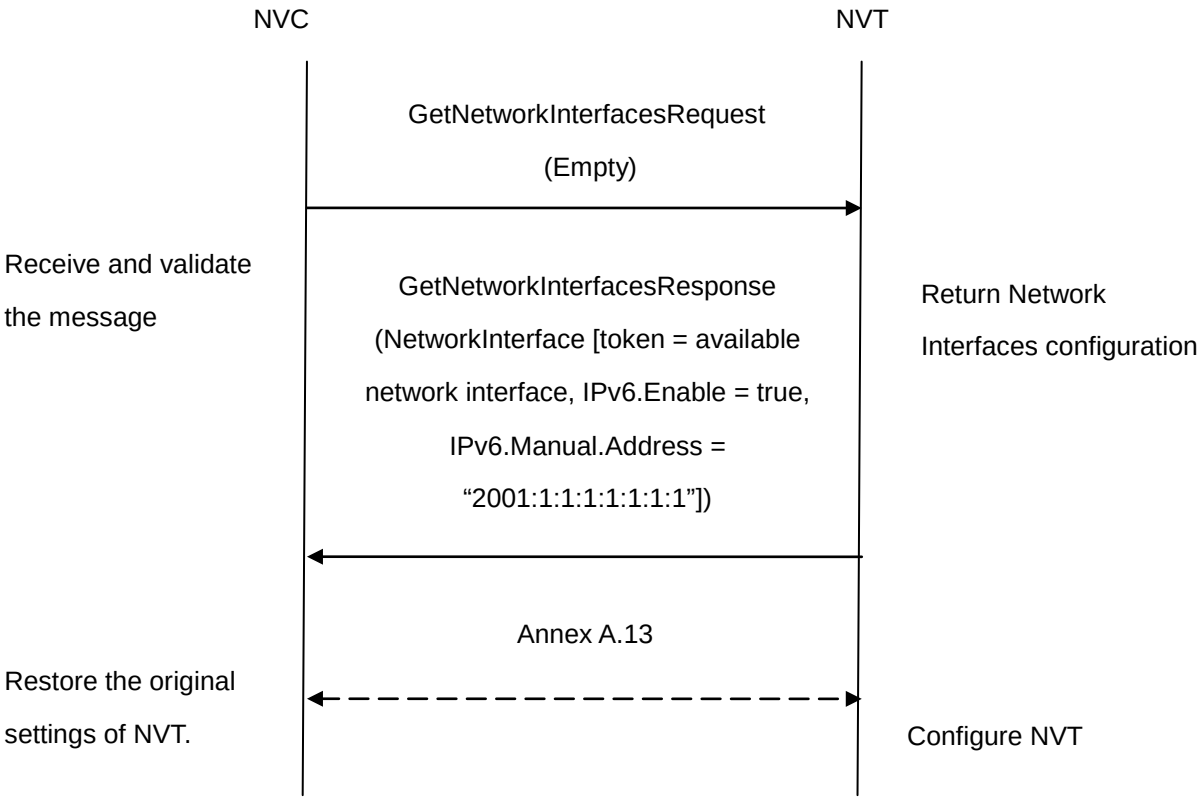
Test Purpose: To configure IPv6 address setting on an NVT through SetNetworkInterfaces command.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
GetNetworkInterfacesRequest
(Empty)
Retrieve the original settings of NVT.
GetNetworkInterfacesResponse
(original settings)
SetNetworkInterfacesRequest
(InterfaceToken = available network interface,
NetworkInterface.IPv6.Enable = true,
NetworkInterface.IPv6.Manual.Address = “2001:1:1:1:1:1:1:1”)
Set IPv6 address to NVT
SetNetworkInterfacesResponse
(RebootNeeded)
Configure NVT Server
Receive and display the message
SystemReboot (if necessary)
(Empty)
Reboot NVT if necessary.
SystemRebootResponse
(Message)
(Reboot in progress)
Multicast Hello Message
(Reboot successful)
Receive and validate Hello Message



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNetworkInterfacesRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetNetworkInterfacesRequest message to set static IPv6 address to NVT (InterfaceToken = available network interface, NetworkInterface.IPv6.Enable = true, NetworkInterface.IPv6.Manual.Address = “2001:1:1:1:1:1:1:1”).
5. NVT will return SetNetworkInterfacesResponse.
6. If necessary, NVC will invoke SystemReboot message to restart NVT. Otherwise, continue to step-8.
7. NVT will return SystemRebootResponse message.
8. NVT will send Multicast Hello message.
9. NVC will receive and validate Hello message sent by NVT.
10. Verify the network interfaces settings in NVT through GetNetworkInterfacesRequest message.

11. NVT sends its network interfaces settings in the GetNetworkInterfacesResponse message (i.e. NetworkInterface[token = available network interface, IPv6.Enable = true, IPv6.Config.Manual = “2001:1:1:1:1:1:1:1”]).
12. NVC will restore the original settings by following the procedure mentioned in Annex A.13.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNetworkInterfacesResponse message.

The DUT did not send SystemRebootResponse message.

The DUT did not send Multicast Hello message after IP configuration change.

The DUT did not send GetNetworkInterfacesResponse message.

The DUT did not send correct network interface information (i.e. NetworkInterface[token = available network interface, IPv6.Enable = true, IPv6.Config.Manual = “2001:1:1:1:1:1:1:1”]) in GetNetworkInterfacesResponse message.

6.2.20 NVT SET NETWORK INTERFACE CONFIGURATION - INVALID IPV4

Test Label: Device Management NVT Network Command SetNetworkInterfaces Test.(for invalid IPv4 address)

ONVIF Core Specification Coverage: 8.2.10 Set network interface configuration

Device Type: NVT

Command Under Test: SetNetworkInterfaces

WSDL Reference: devicemgmt.wsdl

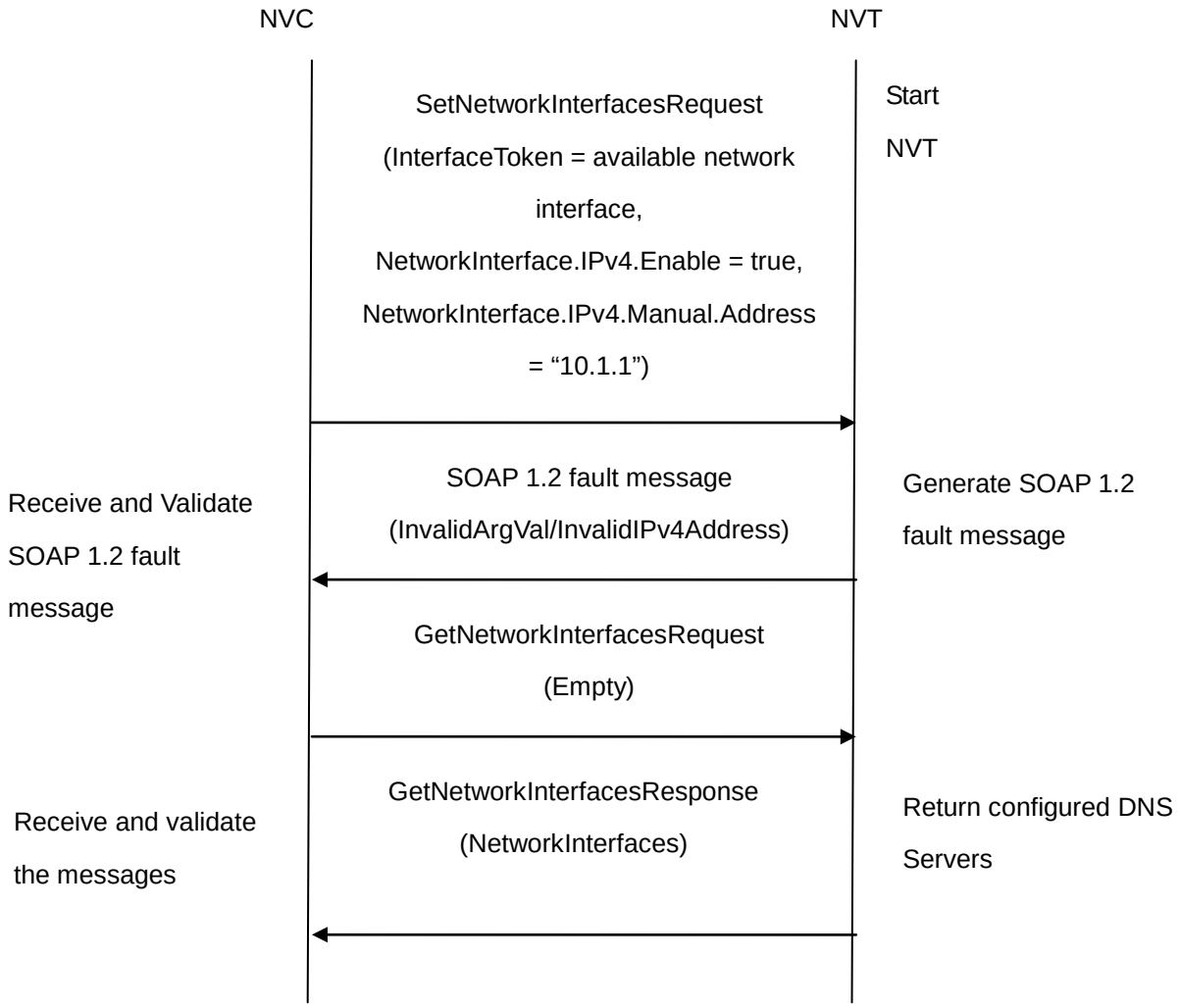
Requirement Level: SHOULD

Test Purpose: To verify behaviour of NVT for invalid IPv4 address configuration.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetDNSRequest message (InterfaceToken = available network interface, NetworkInterface.IPv4.Enable = true, NetworkInterface.IPv4.Manual.Address = “10.1.1”).
4. Verify that the NVT generates SOAP 1.2 fault message (InvalidArgVal/InvalidIPv4Address).
5. Retrieve network interface configurations from NVT through GetNetworkInterfacesRequest message.
6. NVT sends valid network interface configurations in the GetNetworkInterfacesResponse message.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct fault code in the SOAP fault message (InvalidArgVal/InvalidIPv4Address).

The DUT did not send GetNetworkInterfacesResponse message.

The DUT returned “10.1.1” as NVT IPv4 address.

The DUT did not send correct network interface information (i.e. NetworkInterfaces = list of network interfaces) in GetNetworkInterfacesResponse message.

Note: See Annex A.9 for Invalid IPv4 Address and SOAP 1.2 fault message definitions.

6.2.21 NVT SET NETWORK INTERFACE CONFIGURATION - INVALID IPV6

Test Label: Device Management NVT Network Command SetNetworkInterfaces Test.(for invalid IPv6 address)

ONVIF Core Specification Coverage: 8.2.10 Set network interface configuration

Device Type: NVT

Command Under Test: SetNetworkInterfaces

WSDL Reference: devicemgmt.wsdl

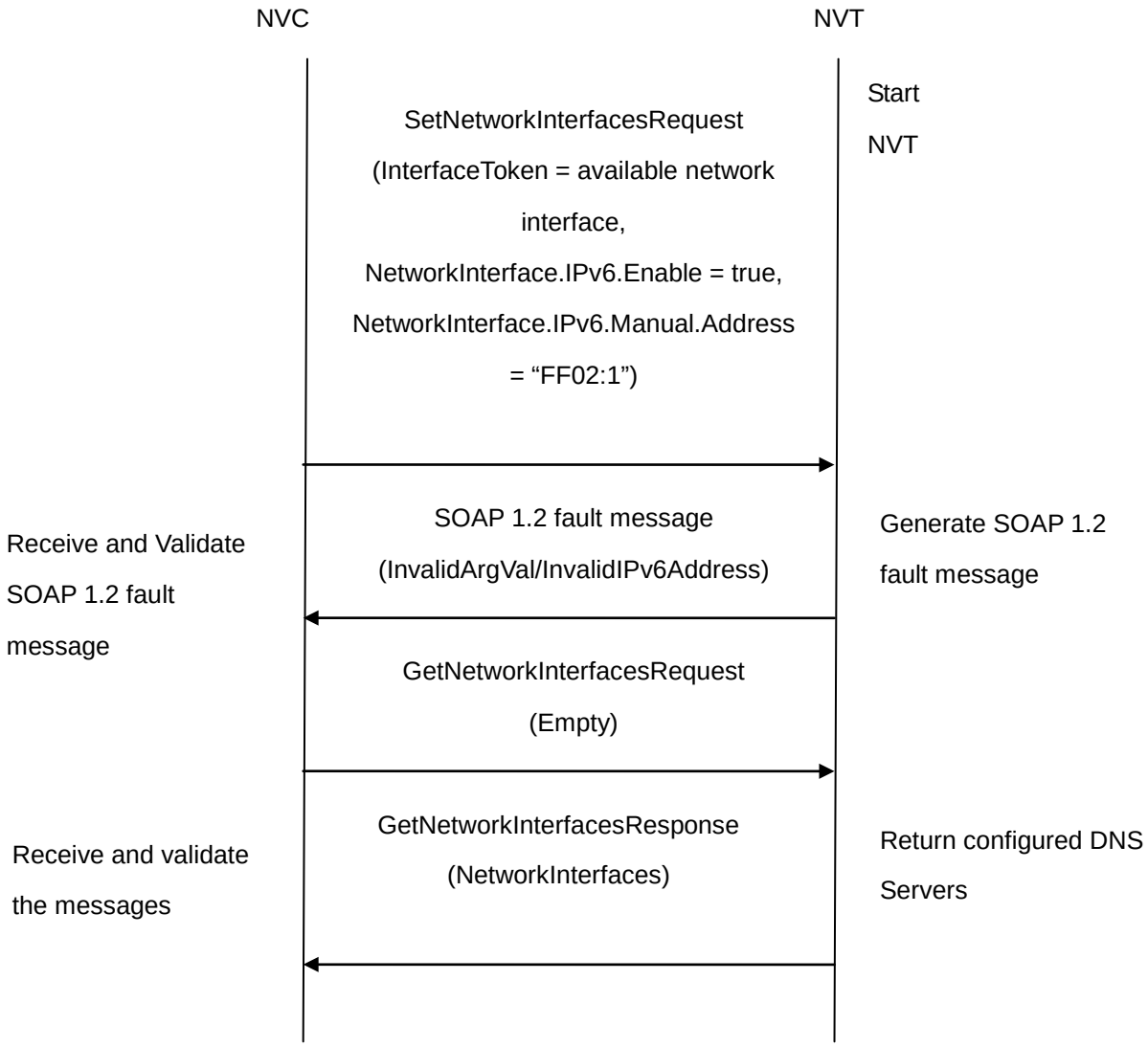
Requirement Level: SHOULD IF IMPLEMENTED (IPv6)

Test Purpose: To verify behaviour of NVT for invalid IPv6 address configuration.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetDNSRequest message (InterfaceToken = available network interface, NetworkInterface.IPv6.Enable = true, NetworkInterface.IPv6.Manual.Address = “FF02:1”).
4. Verify that the NVT generates SOAP 1.2 fault message (InvalidArgVal/InvalidIPv6Address).
5. Retrieve network interface configurations from NVT through GetNetworkInterfacesRequest message.
6. NVT sends valid network interface configurations in the GetNetworkInterfacesResponse message.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct fault code in the SOAP fault message (InvalidArgVal/InvalidIPv6Address).

The DUT did not send GetNetworkInterfacesResponse message.

The DUT returned “FF02:1” as NVT IPv6 address.

The DUT did not send correct network interface information (i.e. NetworkInterfaces = list of network interfaces) in GetNetworkInterfacesResponse message.

6.2.22 NVT GET NETWORK PROTOCOLS CONFIGURATION

Test Label: Device Management NVT Network Command GetNetworkProtocols Test.

ONVIF Core Specification Coverage: 8.2.11 Get network protocols

Device Type: NVT

Command Under Test: GetNetworkProtocols

WSDL Reference: devicemgmt.wsdl

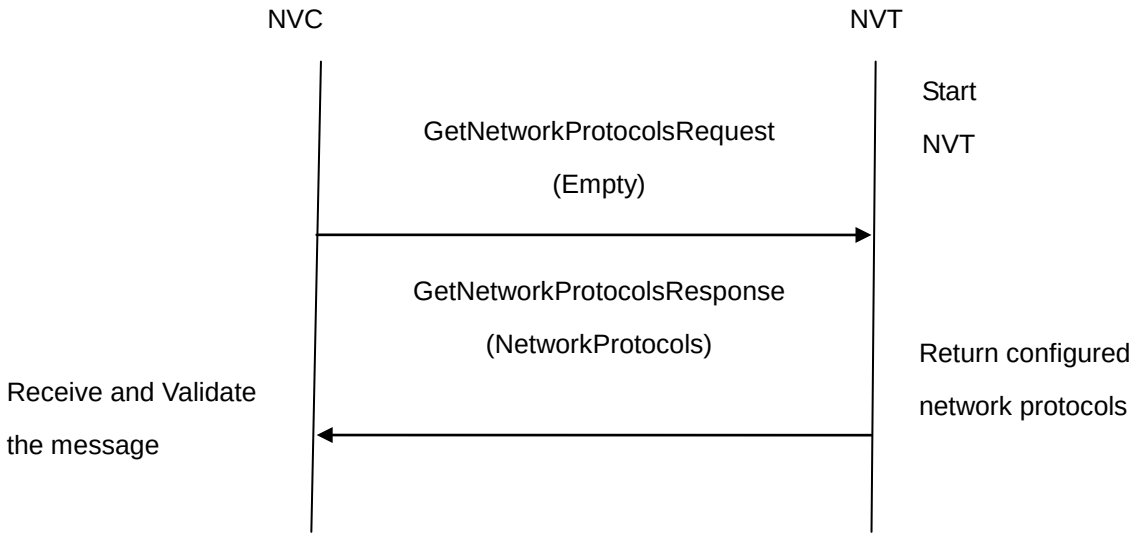
Requirement Level: MUST

Test Purpose: To retrieve network protocols configurations of NVT through GetNetworkProtocols command.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNetworkProtocolsRequest message to retrieve configured network protocols of the NVT.
4. Verify the GetNetworkProtocolsResponse from NVT (NetworkProtocols = list of configured network protocols) and check that the mandatory protocols (RTSP & HTTP) are present in the list.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetNetworkProtocolsResponse message.

The DUT did not send correct information (i.e. NetworkProtocols = list of configured network protocols, the mandatory protocols (RTSP & HTTP) are present in the list) in the GetNetworkProtocolsResponse message.

6.2.23 NVT SET NETWORK PROTOCOLS CONFIGURATION

Test Label: Device Management NVT Network Command SetNetworkProtocols Test.

ONVIF Core Specification Coverage: 8.2.12 Set network protocols

Device Type: NVT

Command Under Test: SetNetworkProtocols

WSDL Reference: devicemgmt.wsdl

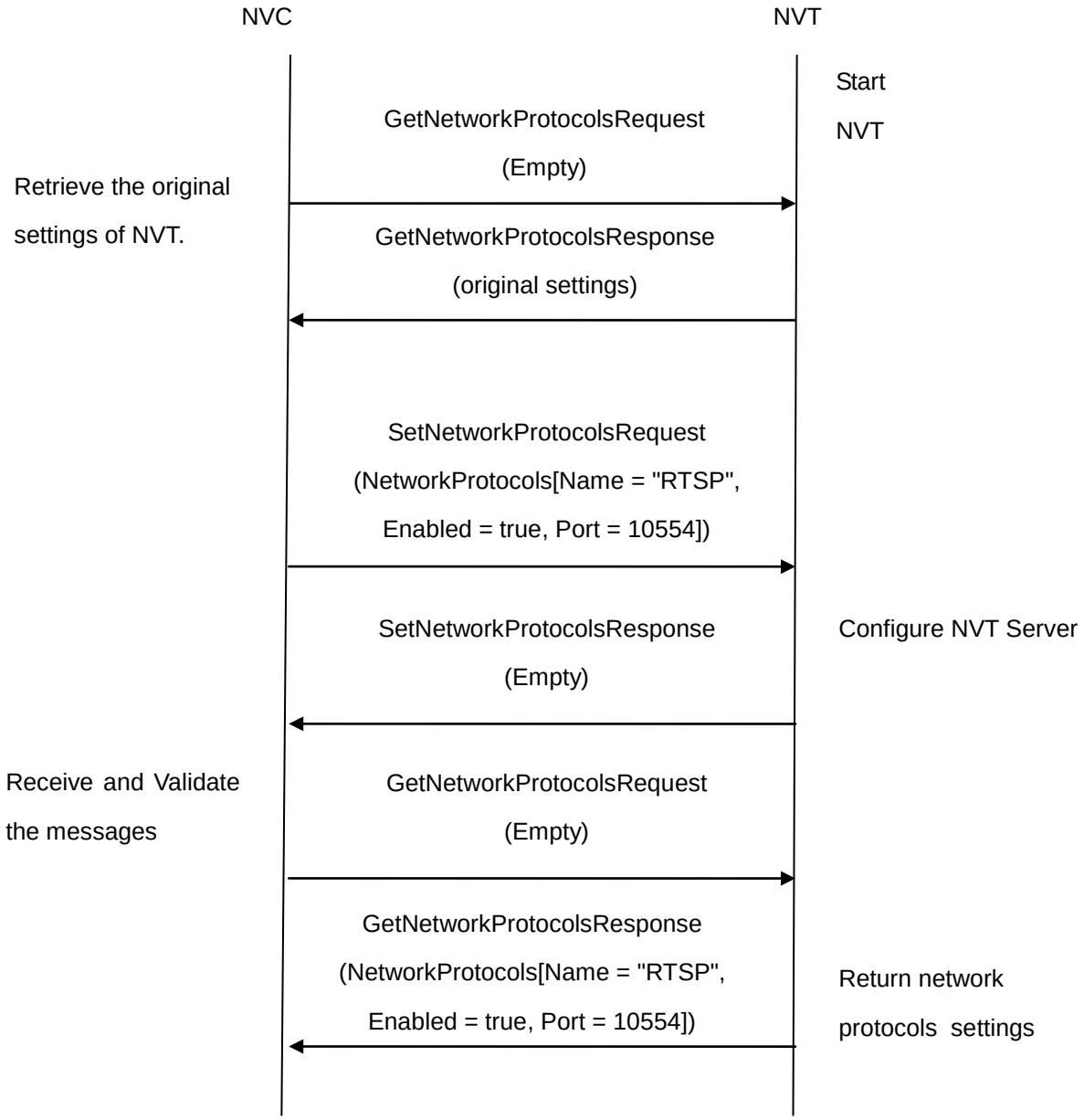
Requirement Level: MUST

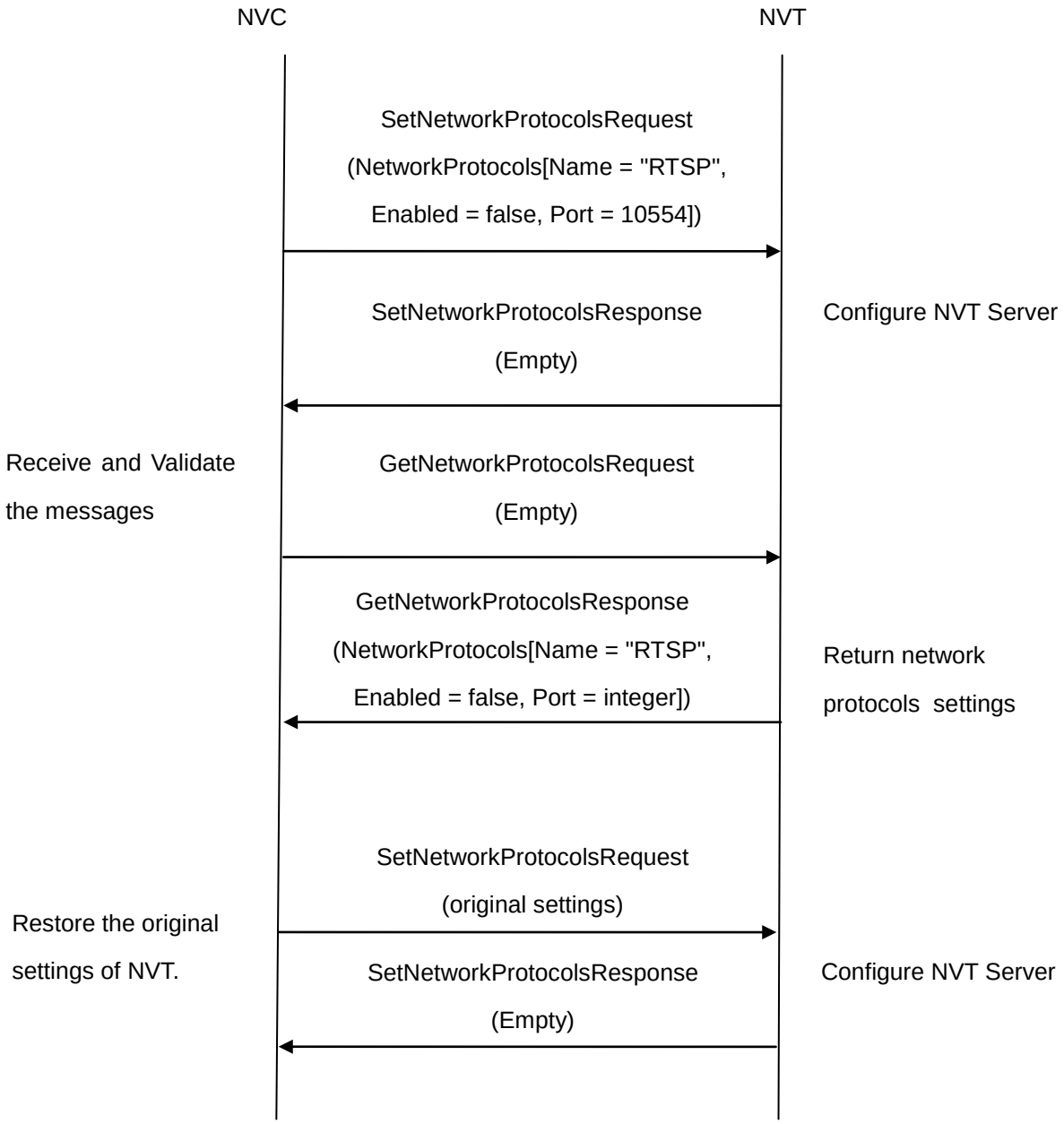
Test Purpose: To configure network protocols setting on an NVT through SetNetworkProtocols command.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start
NVT
GetNetworkProtocolsRequest
(Empty)
Retrieve the original
settings of NVT.
GetNetworkProtocolsResponse
(original settings)
SetNetworkProtocolsRequest
(NetworkProtocols[Name = "RTSP",
Enabled = true, Port = 10554])
SetNetworkProtocolsResponse
(Empty)
Configure NVT Server
Receive and Validate
the messages
GetNetworkProtocolsRequest
(Empty)
GetNetworkProtocolsResponse
(NetworkProtocols[Name = "RTSP",
Enabled = true, Port = 10554])
Return network
protocols settings



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNetworkProtocolsRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetNetworkProtocolsRequest message (NetworkProtocols[Name = "RTSP", Enabled = true, Port = 10554]).
5. Verify that the NVT sends SetNetworkProtocolsResponse (empty message).

6. Verify the network protocols settings in NVT through GetNetworkProtocolsRequest message.
7. NVT sends its network protocols settings in the GetNetworkProtocolsResponse message (NetworkProtocols[Name = "RTSP", Enabled = true, Port = 10554]).
8. NVC will invoke SetNetworkProtocolsRequest message (NetworkProtocols[Name = "RTSP", Enabled = false, Port = 10554]).
9. Verify that the NVT sends SetNetworkProtocolsResponse (empty message).
10. Verify the network protocols settings in NVT through GetNetworkProtocolsRequest message.
11. NVT sends its network protocols settings in the GetNetworkProtocolsResponse message (NetworkProtocols[Name = "RTSP", Enabled = false, Port = integer]).
12. NVC will invoke SetNetworkProtocolsRequest message to restore the original settings of NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNetworkProtocolsResponse message.

The DUT did not send GetNetworkProtocolsResponse message.

The DUT did not send correct network protocols information (i.e. NetworkProtocols[Name = "RTSP", Enabled = true, Port = 10554]) in GetNetworkProtocolsResponse message in step-7.

The DUT did not send correct network protocols information (i.e. NetworkProtocols[Name = "RTSP", Enabled = false, Port = integer]) in GetNetworkProtocolsResponse message in step-11.

6.2.24 NVT SET NETWORK PROTOCOLS CONFIGURATION - UNSUPPORTED PROTOCOLS

Test Label: Device Management NVT Network Command SetNetworkProtocols Test. (for unsupported protocols)

ONVIF Core Specification Coverage: 8.2.12 Set network protocols

Device Type: NVT

Command Under Test: SetNetworkProtocols

WSDL Reference: devicemgmt.wsdl

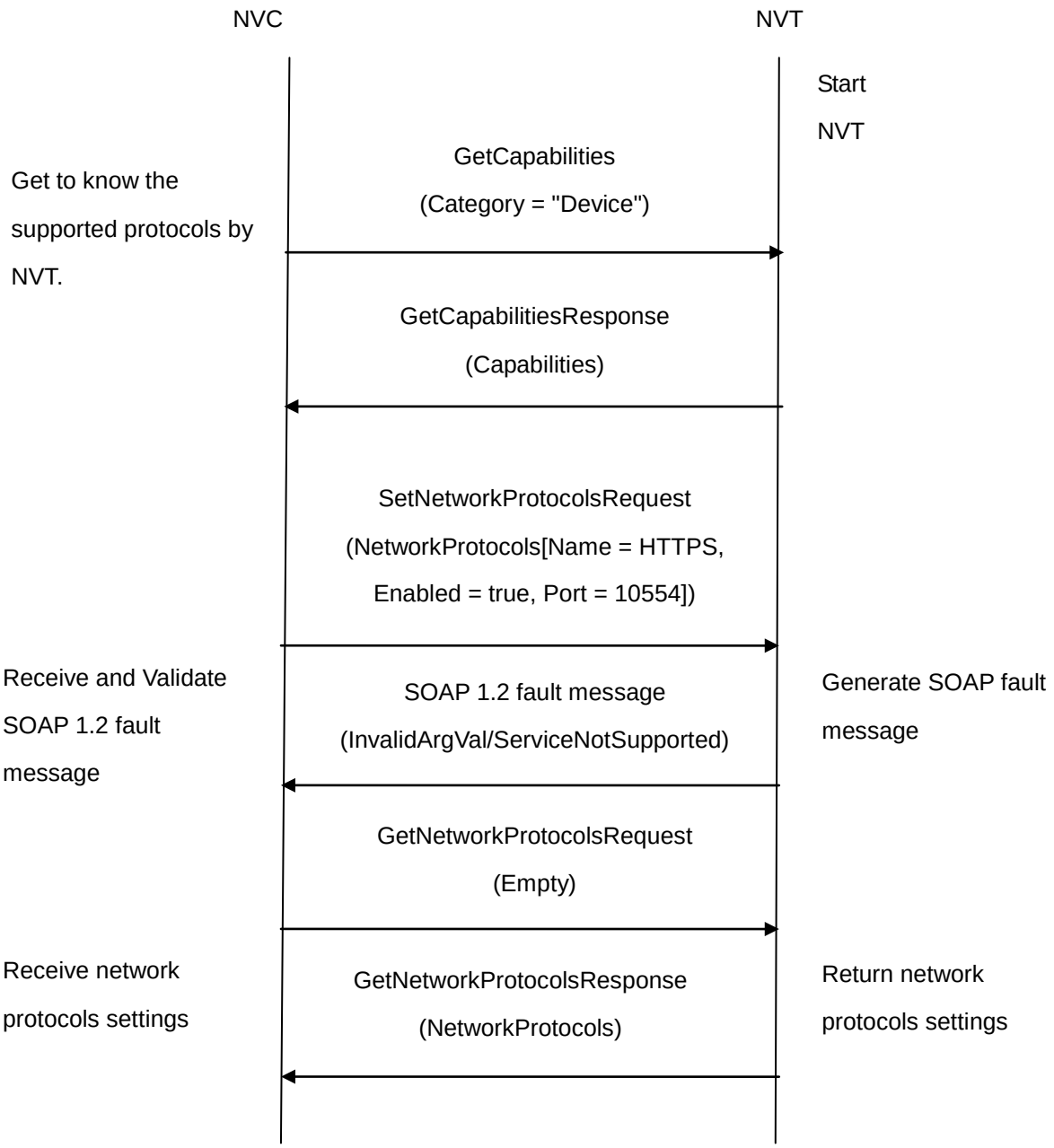
Requirement Level: SHOULD

Test Purpose: To verify behaviour of NVT for unsupported protocols configuration.

Pre-Requisite: NVT does not support all protocols.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetCapabilities message (Category = "Device") to get to know the supported protocols by NVT.

4. NVT will return GetCapabilitiesResponse message.
5. If NVT supports HTTPS (i.e. GetCapabilitiesResponse message includes Capabilities.Device.Security.TLS1.1 = true or Capabilities.Device.Security.TLS1.2 = true), then continue to next test case.
6. NVC will invoke SetNetworkProtocolsRequest message (NetworkProtocols[Name = HTTPS, Enabled = true, Port = 10554]).
7. Verify that the NVT generates SOAP 1.2 fault message (InvalidArgVal/ ServiceNotSupported).
8. Retrieve network protocol configurations from NVT through GetNetworkProtocolsRequest message.
9. NVT sends valid network protocol configurations in the GetNetworkProtocolsResponse message (NetworkProtocols = supported protocols).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct fault code in the SOAP fault message (InvalidArgVal/ ServiceNotSupported).

The DUT did not send GetNetworkProtocolsResponse message.

The DUT returned unsupported protocols in GetNetworkProtocolsResponse message.

The DUT did not send correct network protocols information (i.e. NetworkProtocols = supported protocols) in GetNetworkProtocolsResponse message.

The DUT did not send GetCapabilitiesResponse message.

6.2.25 NVT GET NETWORK DEFAULT GATEWAY CONFIGURATION

Test Label: Device Management NVT Network Command GetNetworkDefaultGateway Test.

ONVIF Core Specification Coverage: 8.2.13 Get default gateway

Device Type: NVT

Command Under Test: GetNetworkDefaultGateway

WSDL Reference: devicemgmt.wsdl

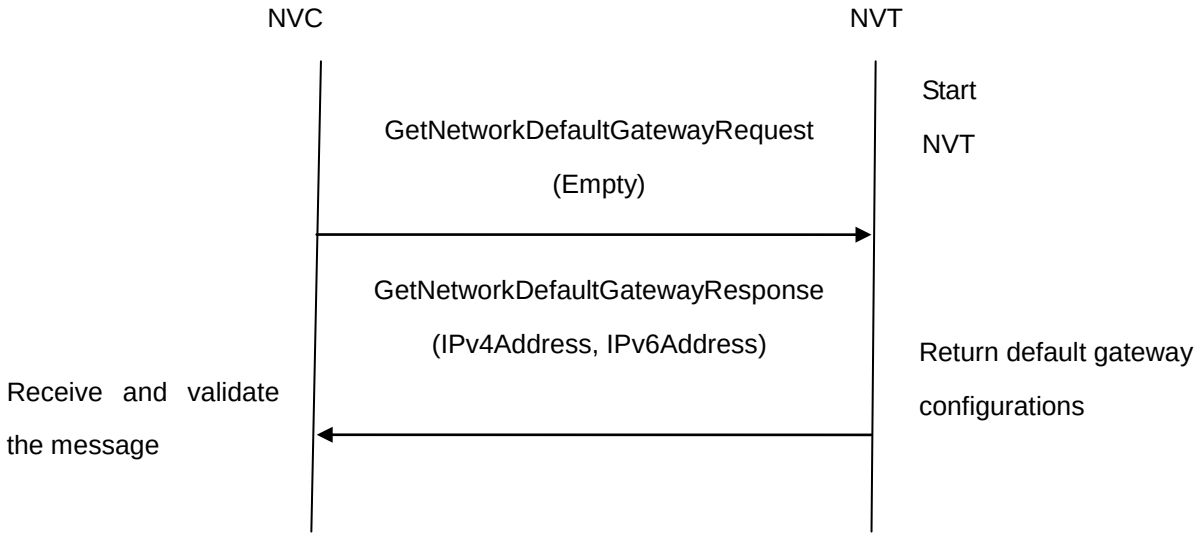
Requirement Level: MUST

Test Purpose: To retrieve default gateway setting of NVT through GetNetworkDefaultGateway command.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNetworkDefaultGatewayRequest message to retrieve default gateway settings of the NVT.
4. Verify the GetNetworkDefaultGatewayResponse from NVT (IPv4Address = list of IPv4 default gateway address, IPv6Address = list of IPv6 default gateway address).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetNetworkDefaultGatewayResponse message.

The DUT did not send correct default gateway information (i.e. IPv4Address = list of IPv4 default gateway address, IPv6Address = list of IPv6 default gateway address) in GetNetworkDefaultGatewayResponse message.

Note: See Annex A.19 for valid expression in terms of empty IP address.

6.2.26 NVT SET NETWORK DEFAULT GATEWAY CONFIGURATION - IPV4

Test Label: Device Management NVT Network Command SetNetworkDefaultGateway Test. (for IPv4 address)

ONVIF Core Specification Coverage: 8.2.14 Set default gateway

Device Type: NVT

Command Under Test: SetNetworkDefaultGateway

WSDL Reference: devicemgmt.wsdl

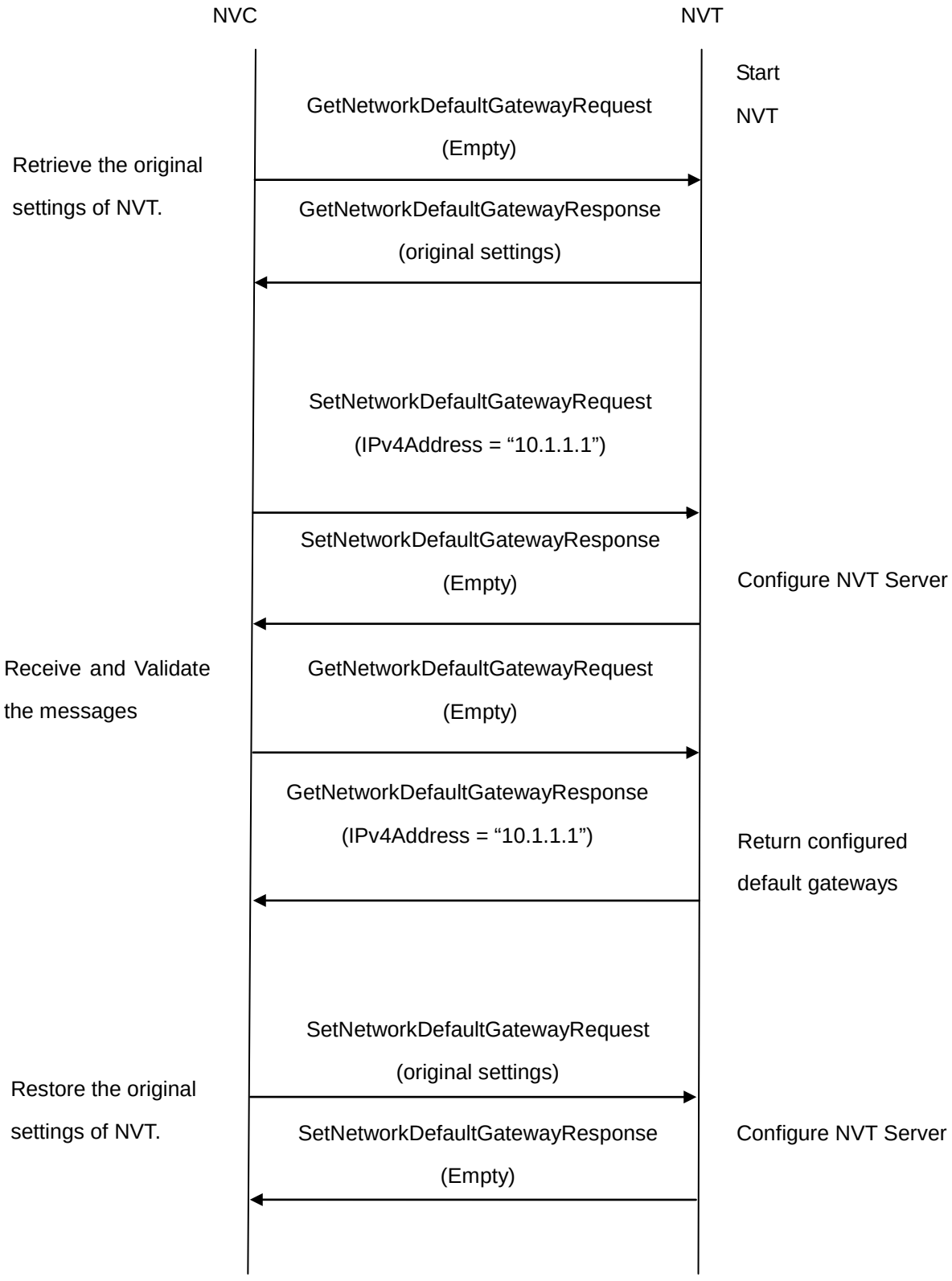
Requirement Level: MUST

Test Purpose: To configure default gateway IPv4 address setting on an NVT through SetNetworkDefaultGateway command.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.

2. Start an NVT.
3. NVC will invoke GetNetworkDefaultGatewayRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetNetworkDefaultGatewayRequest message (IPv4Address = “10.1.1.1”).
5. Verify that the NVT sends SetNetworkDefaultGatewayResponse (empty message).
6. Verify the configured default gateways settings in NVT through GetNetworkDefaultGatewayRequest message.
7. NVT sends its configured default gateways settings in the GetNetworkDefaultGatewayResponse message (IPv4Address = “10.1.1.1”).
8. NVC will invoke SetNetworkDefaultGatewayRequest message to restore the original settings of NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNetworkDefaultGatewayResponse message in step-5.

The DUT did not send GetNetworkDefaultGatewayResponse message in step-7.

The DUT did not send correct default gateway information (i.e. IPv4Address = “10.1.1.1”) in GetNetworkDefaultGatewayResponse message in step-7.

Note: See Annex A.9 for Valid IPv4 Address definition.

See Annex A.19 for valid expression in terms of empty IP address.

6.2.27 NVT SET NETWORK DEFAULT GATEWAY CONFIGURATION - IPV6

Test Label: Device Management NVT Network Command SetNetworkDefaultGateway Test. (for IPv6 address)

ONVIF Core Specification Coverage: 8.2.14 Set default gateway

Device Type: NVT

Command Under Test: SetNetworkDefaultGateway

WSDL Reference: devicemgmt.wsdl

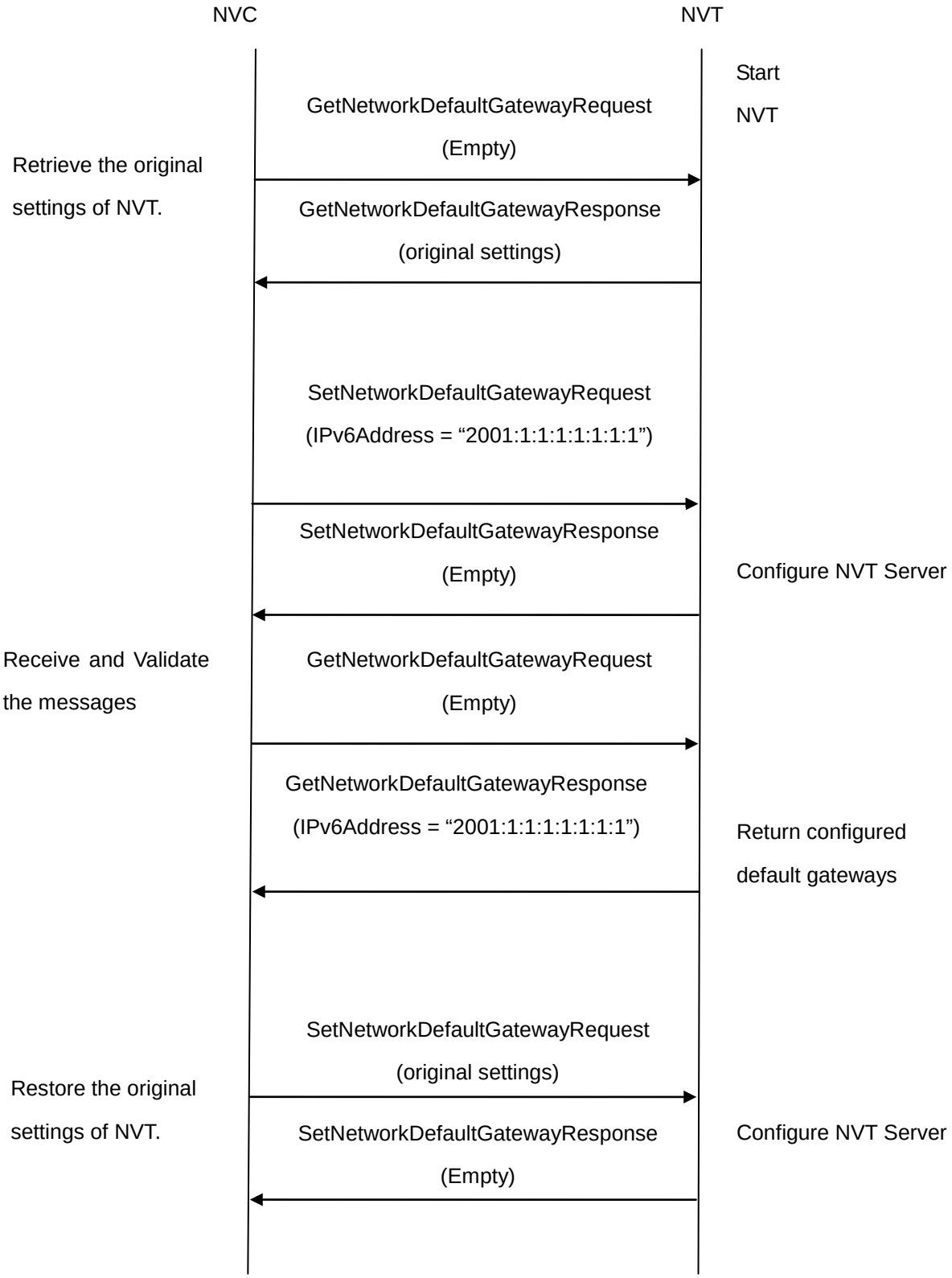
Requirement Level: SHOULD IF IMPLEMENTED (IPv6)

Test Purpose: To configure default gateway IPv6 address setting on an NVT through SetNetworkDefaultGateway command.

Pre-Requisite: IPv6 is implemented by NVT.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNetworkDefaultGatewayRequest message to retrieve the original settings of NVT.
4. NVC will invoke SetNetworkDefaultGatewayRequest message (IPv6Address = “2001:1:1:1:1:1:1:1”).
5. Verify that the NVT sends SetNetworkDefaultGatewayResponse (empty message).
6. Verify the configured default gateways settings in NVT through GetNetworkDefaultGagewayRequest message.
7. NVT sends its configured default gateways settings in the GetNetworkDefaultGatewayResponse message (IPv6Address = “2001:1:1:1:1:1:1:1”).
8. NVC will invoke SetNetworkDefaultGatewayRequest message to restore the original settings of NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetNetworkDefaultGatewayResponse message in step-5.

The DUT did not send GetNetworkDefaultGatewayResponse message in step-7.

The DUT did not send correct default gateway information (i.e. IPv6Address = “2001:1:1:1:1:1:1:1”) in GetNetworkDefaultGatewayResponse message in step-7.

6.2.28 NVT SET NETWORK DEFAULT GATEWAY CONFIGURATION - INVALID IPV4

Test Label: Device Management NVT Network Command SetNetworkDefaultGateway Test. (for invalid IPv4 address)

ONVIF Core Specification Coverage: 8.2.14 Set default gateway

Device Type: NVT

Command Under Test: SetNetworkDefaultGateway

WSDL Reference: devicemgmt.wsdl

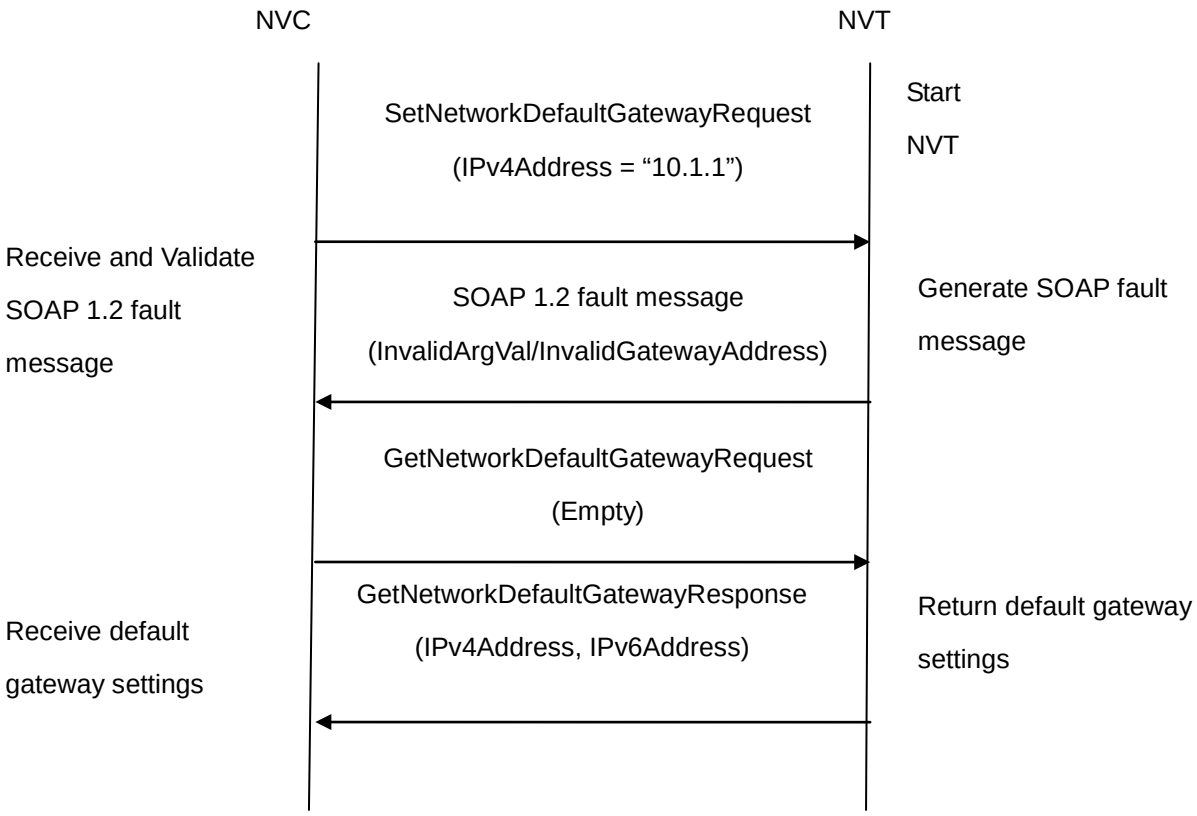
Requirement Level: SHOULD

Test Purpose: To verify behaviour of NVT for invalid default gateway IPv4 address configuration.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetNetworkDefaultGatewayRequest message (IPv4Address = “10.1.1”).
4. Verify that the NVT generates SOAP 1.2 fault message (InvalidArgVal/InvalidGatewayAddress).
5. Retrieve default gateway configurations from NVT through GetNetworkDefaultGatewayRequest message.
6. NVT sends valid default gateway configurations in the GetNetworkDefaultGatewayResponse message (IPv4Address = list of IPv4 default gateway address, IPv6Address = list of IPv6 default gateway addresses).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct fault code in the SOAP fault message (InvalidArgVal/InvalidGatewayAddress).

The DUT did not GetNetworkDefaultGatewayResponse message.

The DUT returned “10.1.1” as IPv4 default gateway address.

The DUT did not send correct default gateway information (i.e. IPv4Address = list of IPv4 default gateway address, IPv6Address = list of IPv6 default gateway addresses) in GetNetworkDefaultGatewayResponse message.

Note: See Annex A.9 for Invalid IPv4 Address and SOAP 1.2 fault message definitions.

6.2.29 NVT SET NETWORK DEFAULT GATEWAY CONFIGURATION - INVALID IPV6

Test Label: Device Management NVT Network Command SetNetworkDefaultGateway Test.(for invalid IPv6 address)

ONVIF Core Specification Coverage: 8.2.14 Set default gateway

Device Type: NVT

Command Under Test: SetNetworkDefaultGateway

WSDL Reference: devicemgmt.wsdl

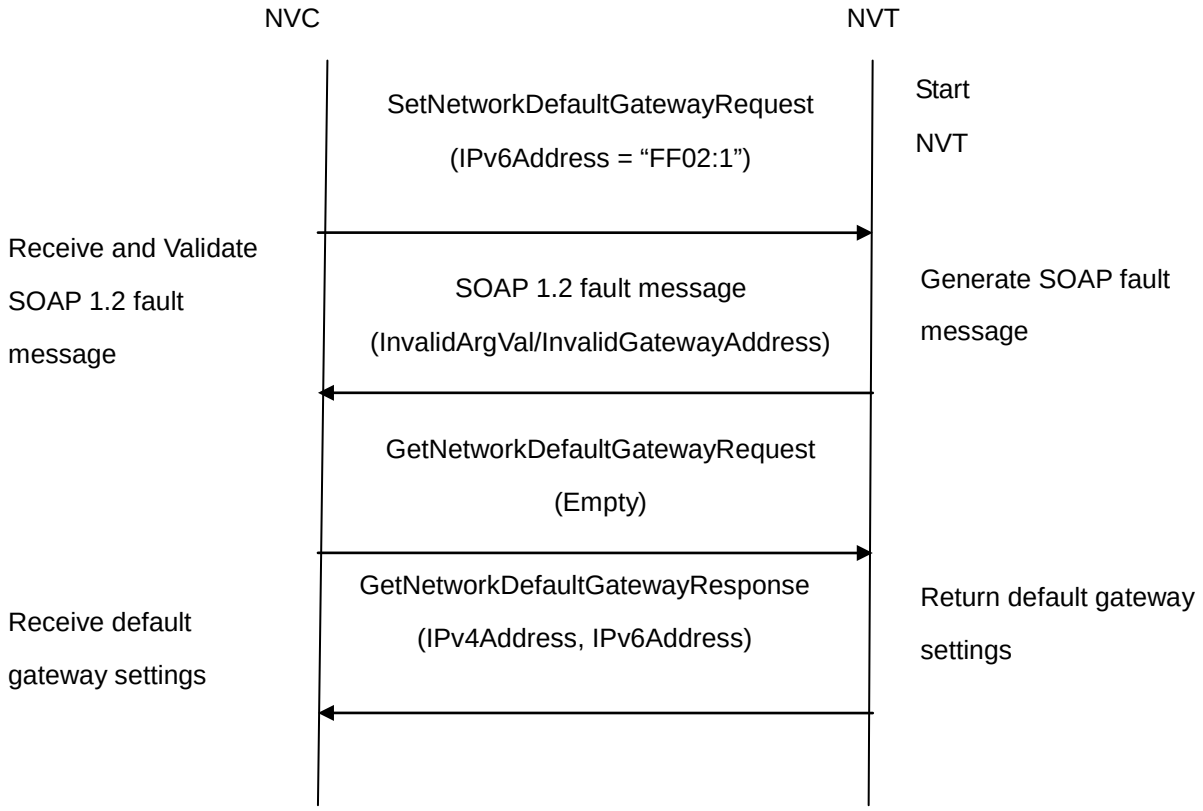
Requirement Level: SHOULD IMPLEMENTED (IPv6)

Test Purpose: To verify behaviour of NVT for invalid default gateway IPv6 address configuration.

Pre-Requisite: IPv6 is implemented by NVT.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetNetworkDefaultGatewayRequest message (IPv6Address = “FF02:1”).
4. Verify that the NVT generates SOAP 1.2 fault message (InvalidArgVal/InvalidGatewayAddress).
5. Retrieve default gateway configurations from NVT through GetNetworkDefaultGatewayRequest message.
6. NVT sends valid default gateway configurations in the GetNetworkDefaultGatewayResponse message (IPv4Address = list of IPv4 default gateway address, IPv6Address = list of IPv6 default gateway addresses).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct fault code in the SOAP fault message (InvalidArgVal/InvalidGatewayAddress).

The DUT did not GetNetworkDefaultGatewayResponse message.

The DUT returned “FF02:1” as IPv6 default gateway address.

The DUT did not send correct default gateway information (i.e. IPv4Address = list of IPv4 default gateway address, IPv6Address = list of IPv6 default gateway addresses) in GetNetworkDefaultGatewayResponse message.

6.3 System

6.3.1 NVT SYSTEM COMMAND GETSYSTEMDATEANDTIME

Test Label: Device Management NVT System Command GetSystemDateAndTime Test.

ONVIF Core Specification Coverage: 8.3.4 Get system date and time

Device Type: NVT

Command Under Test: GetSystemDateAndTime

Requirement Level: MUST

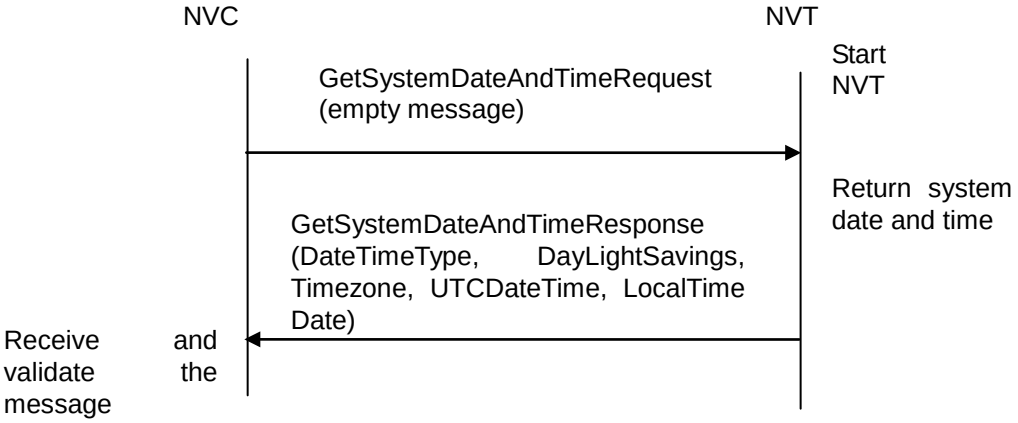
WSDL Reference: devicemgmt.wsdl

Test Purpose: To retrieve NVT system date and time through GetSystemDateAndTime command.

Pre-Requisite: None.

Test Configuration: NVC and NVT.

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetSystemDateAndTimeRequest message to get NVT system date and time.
4. Verify system date and time configurations of NVT in GetSystemDateAndTimeResponse message (DateTimeType = Manual or NTP, DayLightSavings = true or false, Timezone = POSIX 1003.1, UTC DateTime = Hour:Min:Sec, Year:Month:Day and LocalTimeDate = Hour:Min:Sec, Year:Month:Day).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetSystemDateAndTimeResponse message.

The DUT did not send DateTimeType and DayLightSavings information in the GetSystemDateAndTimeResponse message.

Note: If system date and time are set manually, then DUT MUST return UTCDateTime or LocalDateTime in the GetSystemDateAndTimeResponse message.

6.3.2 NVT SYSTEM COMMAND SETSYSTEMDATEANDTIME

Test Label: Device Management NVT Network Command SetSystemDateAndTime Test.

ONVIF Core Specification Coverage: 8.3.5 Set system date and time

Device Type: NVT

Command Under Test: SetSystemDateAndTime

Requirement Level: MUST

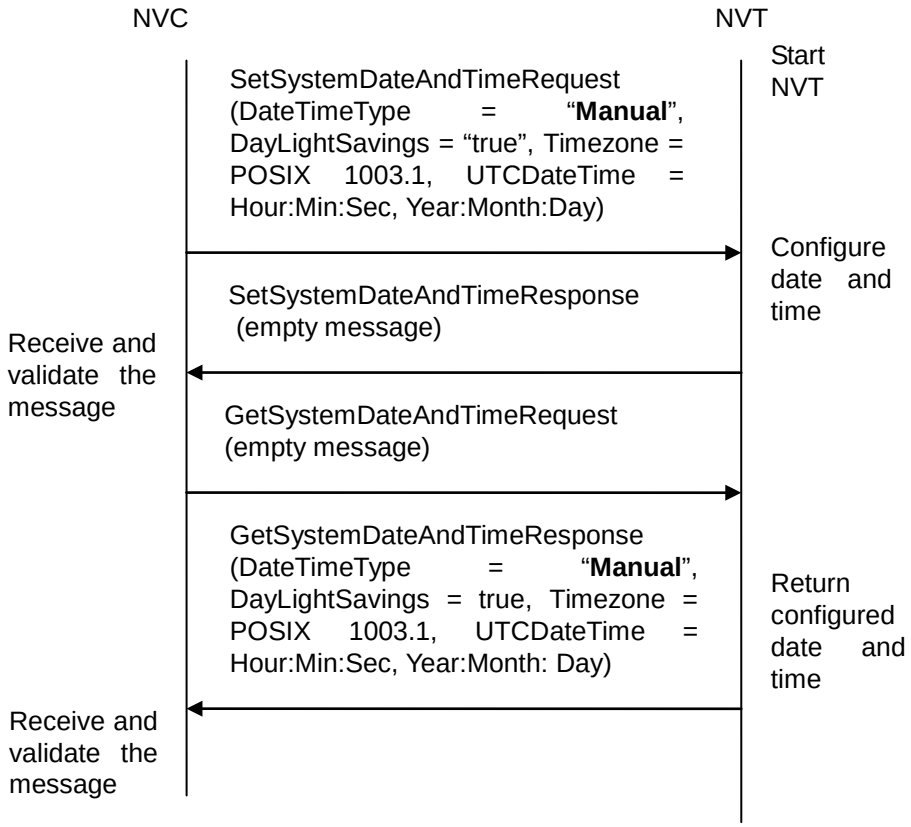
WSDL Reference: devicemgmt.wsdl

Test Purpose: To set the NVT system date and time using SetSystemDateAndTime command, date and time is entered manually.

Pre-Requisite: None.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetSystemDateAndTimeRequest message (DateTimeType = "Manual", DayLightSavings = true, Timezone = POSIX

   1003.1, UTCDateTime = Hour:Min:Sec, Year:Month:Day) to configure the date and time in the NVT.
4. Verify that NVT sends SetSystemDateAndTimeResponse message (empty message).
5. Verify the NVT date and time configurations through GetSystemDateAndTimeRequest message.
6. NVT sends system date and time configurations in the GetSystemDateAndTimeResponse message (DateTimeType = "Manual", DayLightSavings = true, Timezone = POSIX 1003.1, UTCDateTime = Hour:Min:Sec, Year:Month:Day).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetSystemDateAndTimeResponse message.

The DUT did not send GetSystemDateAndTimeResponse message.

The DUT did not send expected system date and time configuration (DateTimeType = “Manual”, DayLightSavings = true**,** Timezone = POSIX 1003.1, UTCDateTime = Hour:Min:Sec, Year:Month:Day) in the GetSystemDateAndTimeResponse message.

6.3.3 NVT SYSTEM COMMAND SETSYSTEMDATEANDTIME USING NTP

Test Label: Device Management NVT Network Command SetSystemDateAndTime Test using NTP.

ONVIF Core Specification Coverage: 8.3.5 Set system date and time

Device Type: NVT

Command Under Test: SetSystemDateAndTime

Requirement Level: MUST IF SUPPORTED(NTP).

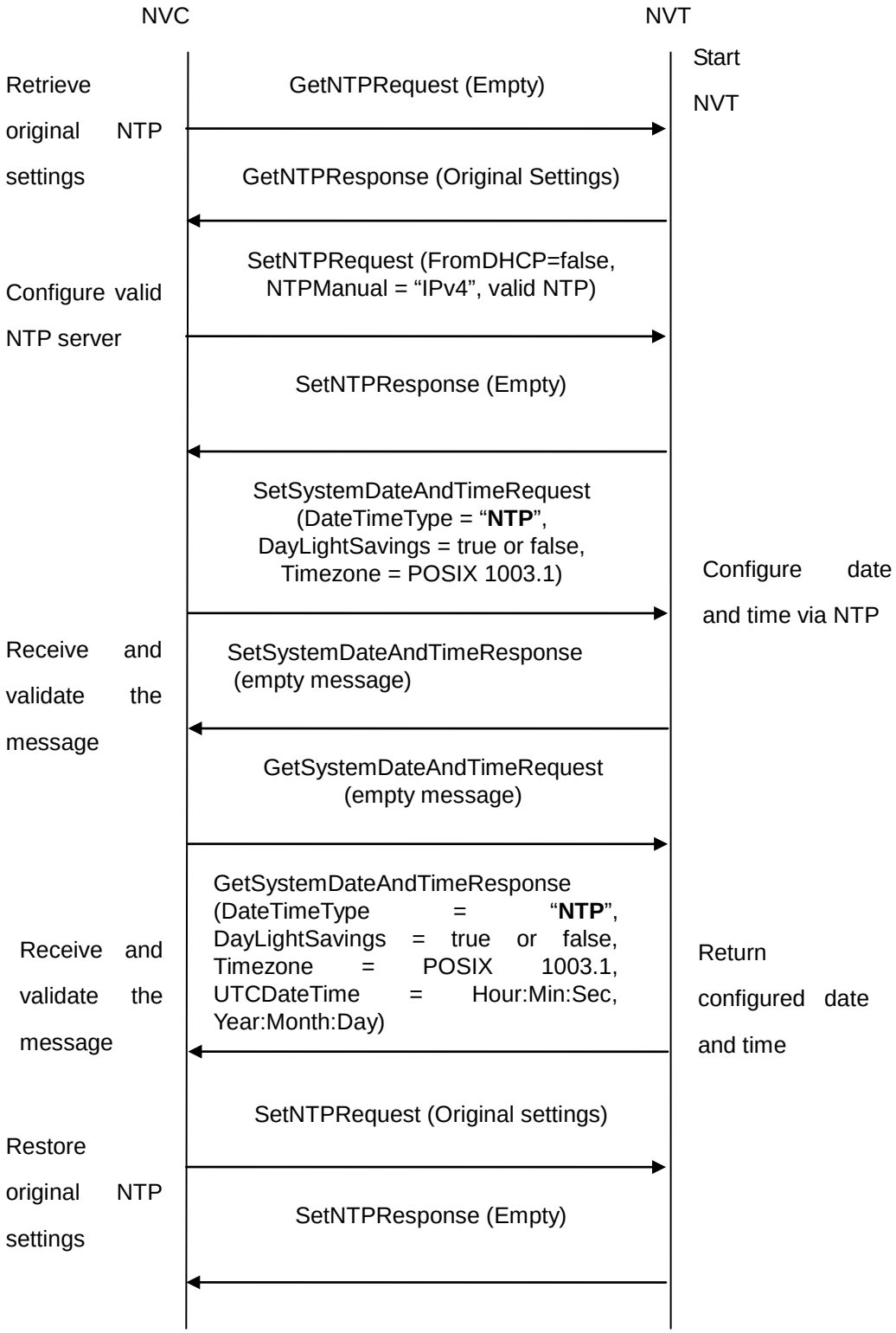
WSDL Reference: devicemgmt.wsdl

Test Purpose: To set the NVT system date and time using **SetSystemDateAndTime** command through NTP.

Pre-Requisite: A valid NTP server address should be configured in the NVT.

Test Configuration: NVC, NVT and NTP.

Test Sequence:



Test Procedure:

1. Start an NVC.

2. Start an NVT.
3. NVC invokes GetNTPRequest to retrieve original NTP settings of NVT.
4. NVC invokes SetNTPRequest (FromDHCP=false, NTPManual = “IPv4”, valid NTP server address) to configure NVT with proper NTP server.
5. NVC will invoke SetSystemDateAndTimeRequest message (DateTimeType = “NTP”, DayLightSavings = true or false, Timezone = POSIX 1003.1) to configure the time in the NVT.
6. NVT shall obtain and configure time via NTP.
7. Verify that the NVT sends SetSystemDateAndTimeResponse (empty message).
8. Verify the NVT date and time configurations through GetSystemDateAndTimeRequest message.
9. NVT sends system date and time configurations in the GetSystemDateAndTimeResponse message (DateTimeType = “NTP”, DayLightSavings = true or false, Timezone = POSIX 1003.1, UTCDateTime = Hour:Min:Sec, Year:Month:Day).
10. NVC invokes SetNTPRequest (original NTP Settings) to restore NTP settings of NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetSystemDateAndTimeResponse message.

The DUT did not send GetSystemDateAndTimeResponse message.

The DUT did not send expected system date and time configuration (DateTimeType = “NTP”, DayLightSavings = true or false**,** Timezone = POSIX 1003.1, UTCDateTime = Hour:Min:Sec**,** Year:Month:Day) in the GetSystemDateAndTimeResponse message.

6.3.4 NVT SYSTEM COMMAND SETSYSTEMDATEANDTIME TEST FOR INVALID TIMEZONE

Test Label: Device Management NVT Network Command **SetSystemDateAndTime** Test, for invalid timezone.

ONVIF Core Specification Coverage: 8.3.5 Set system date and time.

Device Type: NVT

Command Under Test: SetSystemDateAndTime

Requirement Level: SHOULD

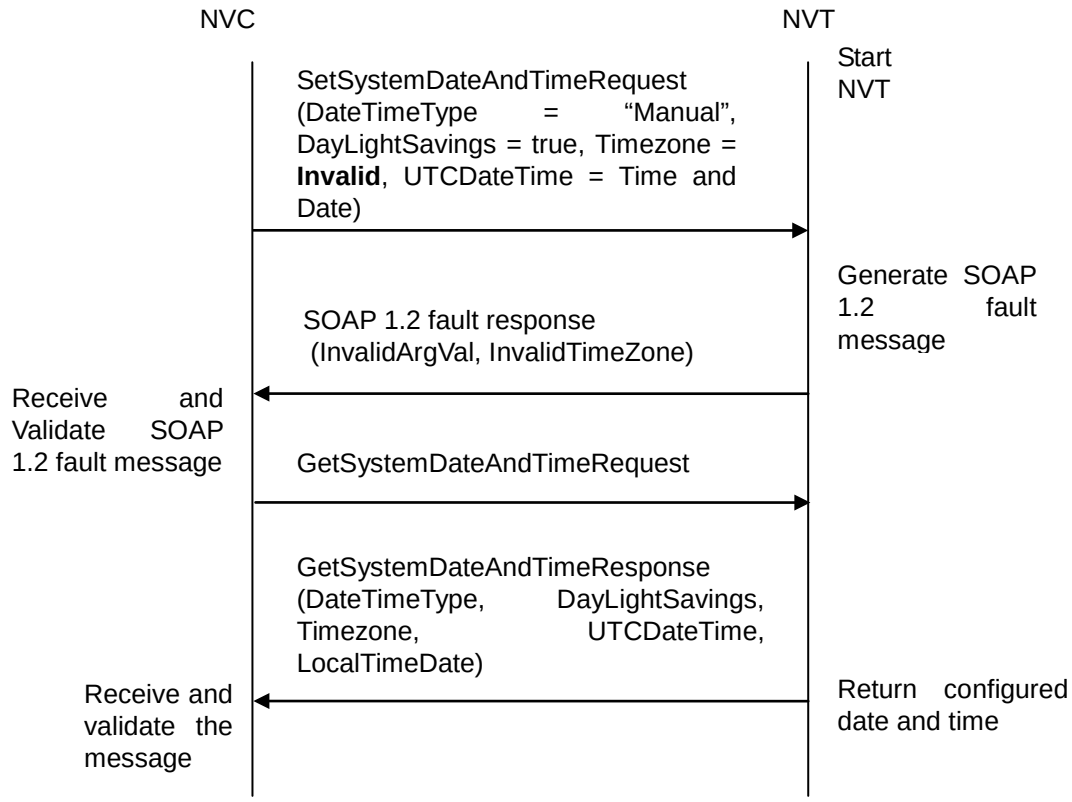
WSDL Reference: devicemgmt.wsdl

Test Purpose: To verify the behaviour of the NVT for invalid Timezone configuration.

Pre-Requisite: None.

Test Configuration: NVC and NVT.

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetSystemDateAndTimeRequest message with invalid Timezone (DateTimeType "Manual", DayLightSavings = "True", Timezone = Invalid, UTCDateTime = Hour:Min:Sec, Year:Month:Day).
4. Verify that NVT generates SOAP 1.2 fault response (InvalidArgVal, InvalidTimeZone).
5. Verify the NVT system date and time configurations through GetSystemDateAndTimeRequest message.
6. NVT sends system date and time configurations in the GetSystemDateAndTimeResponse message (DateTimeType = Manual or NTP, DayLightSavings = true or false, Timezone = POSIX 1003.1, UTC DateTime = Hour:Min:Sec, Year:Month:Day and LocalTimeDate = Hour:Min:Sec, Year:Month:Day).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct fault code in the SOAP fault message (InvalidArgVal, InvalidTimeZone).

The DUT did not send GetSystemDateAndTimeResponse message.

The DUT returned “Invalid Timezone” in the GetSystemDateAndTimeResponse message.

Note: If system date and time are set manually, then DUT MUST return UTCDateTime or LocalDateTime in the GetSystemDateAndTimeResponse message.

See Annex A.4 for Invalid TimeZone and SOAP 1.2 fault message definitions.

6.3.5 NVT SYSTEM COMMAND SETSYSTEMDATEANDTIME TEST FOR INVALID DATE

Test Label: Device Management NVT Network Command **SetSystemDateAndTime** Test, for invalid date and time.

ONVIF Core Specification Coverage: 8.3.5 Set system date and time

Device Type: NVT

Command Under Test: SetSystemDateAndTime

Requirement Level: SHOULD

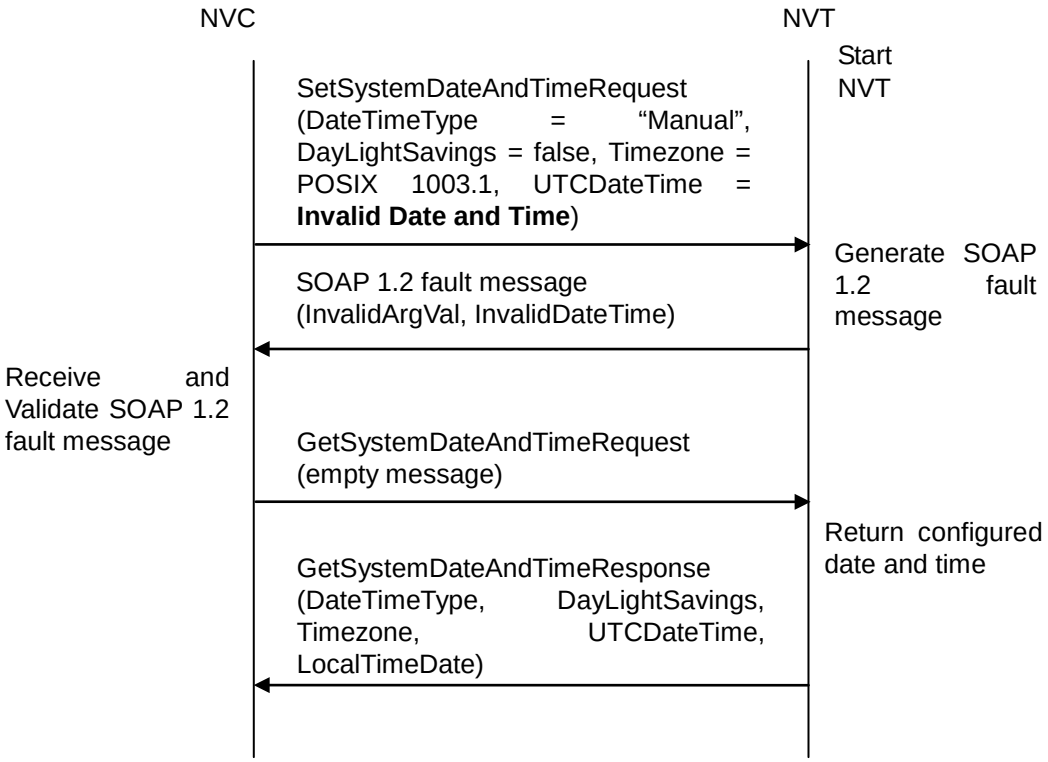
WSDL Reference: devicemgmt.wsdl

Test Purpose: To verify the behaviour of NVT for invalid system date and time configuration.

Pre-Requisite: None.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetSystemDateAndTimeRequest message with invalid Date and Time (DateTimeType = "Manual", DayLightSavings = false, Timezone = POSIX 1003.1, UTCDateTime = Invalid Date and Time).
4. Verify that NVT generates SOAP 1.2 fault message (InvalidArgVal, InvalidDateTime).
5. Verify the NVT system date and time configurations through GetSystemDateAndTimeRequest message.
6. NVT sends system date and time configurations in the GetSystemDateAndTimeResponse message (DateTimeType = Manual or NTP, DayLightSavings = true or false, Timezone = POSIX 1003.1, UTC DateTime = Hour:Min:Sec, Year:Month:Day and LocalTimeDate = Hour:Min:Sec, Year:Month:Day).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct fault code in the SOAP fault message (InvalidArgVal, InvalidDateTime).

The DUT did not send GetSystemDateAndTimeResponse message.

The DUT did not send SetSystemDateAndTimeResponse message.

The DUT returned “Invalid Date and Time” in the GetSystemDateAndTimeResponse message.

Note: If system date and time are set manually, then DUT MUST return UTCDateTime or LocalDateTime in the GetSystemDateAndTimeResponse message.

See Annex A.6 for Invalid SOAP 1.2 fault message definition.

6.3.6 NVT SYSTEM COMMAND FACTORY DEFAULT HARD

Test Label: Device Management NVT System Command **SetSystemFactoryDefault** Test, Hard Reset.

ONVIF Core Specification Coverage: 8.3.6 Factory default

Device Type: NVT

Command Under Test: SetSystemFactoryDefault

Requirement Level: MUST

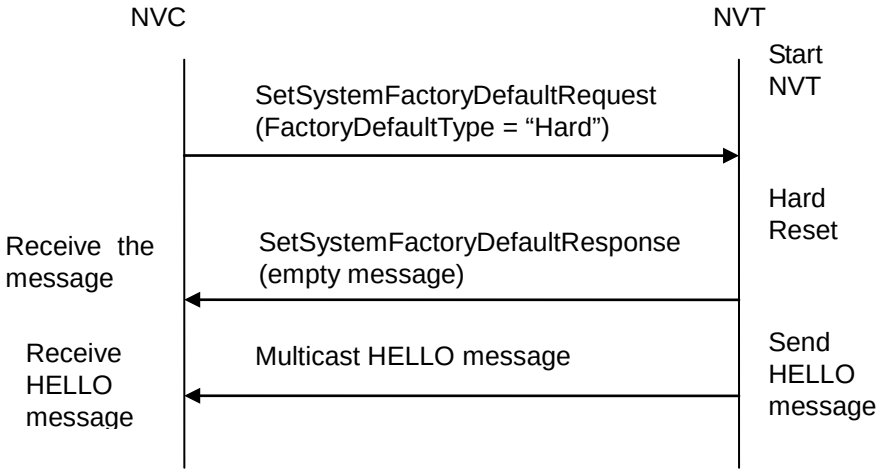
WSDL Reference: devicemgmt.wsdl

Test Purpose: To reload all parameters of NVT to their default values through SetSystemFactoryDefault command. This test is for hard factory default.

Pre-Requisite: None.

Test Configuration: NVC and NVT.

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.

3. NVC will invoke SetSystemFactoryDefaultRequest message (FactoryDefaultType = “Hard”).
4. Verify that NVT sends SetSystemFactoryDefaultResponse message.
5. Verify that NVT sends Multicast HELLO message after hard reset.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetSystemFactoryDefaultResponse message.

The DUT did not send HELLO message.

Note: After Hard Reset certain DUTs are not IP reachable. In such situation, DUT must be configured with an IPv4 address, must be IP reachable in the test network and other relevant configurations to be done for further tests.

6.3.7 NVT SYSTEM COMMAND FACTORY DEFAULT SOFT

Test Label: Device Management NVT System Command **SetSystemFactoryDefault** Test, Soft Reset.

ONVIF Core Specification Coverage: 8.3.6 Factory default

Device Type: NVT

Command Under Test: SetSystemFactoryDefault

Requirement Level: MUST

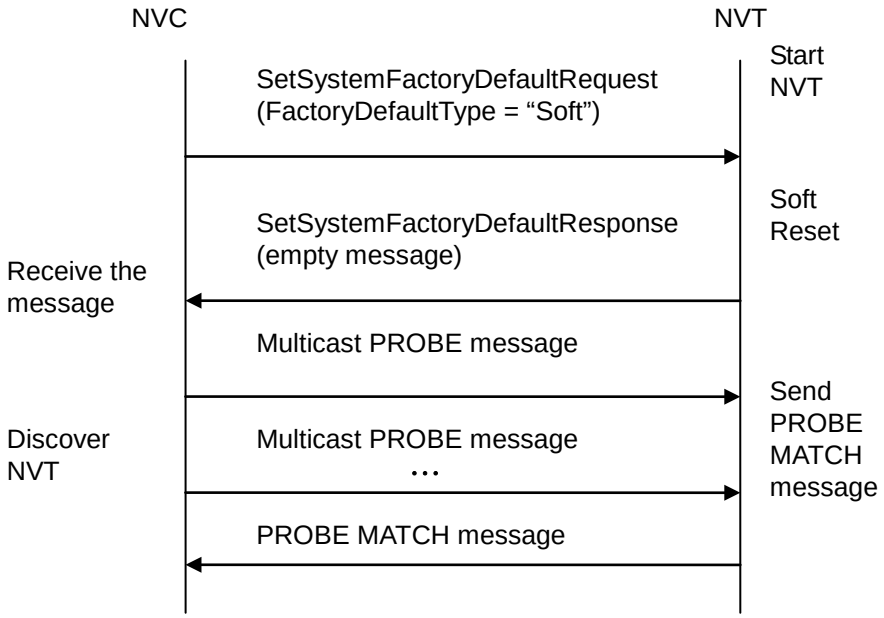
WSDL Reference: devicemgmt.wsdl

Test Purpose: To reload all parameters of NVT to their default values through SetSystemFactoryDefault command. This test is for soft factory default.

Pre-Requisite: None.

Test Configuration: NVC and NVT.

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetSystemFactoryDefaultRequest message (FactoryDefaultType = “Soft”).
4. Verify that NVT sends SetSystemFactoryDefaultResponse message.
5. NVC will verify that NVT is accessible after soft reset. NVC will send Multicast PROBE message several times (i.e. 50 times at an interval of 5 seconds).
6. Verify that NVT sends a PROBE MATCH message.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SetSystemFactoryDefaultResponse message.

The DUT did not send PROBE MATCH message (i.e. DUT cannot be discovered).

Note: After a soft reset some DUTs require some configurations to be done for further tests.

6.3.8 NVT SYSTEM COMMAND REBOOT

Test Label: Device Management NVT System Command SystemReboot Test.

ONVIF Core Specification Coverage: 8.3.10 Reboot

Command Under Test: SystemReboot

Device Type: NVT

Requirement Level: MUST

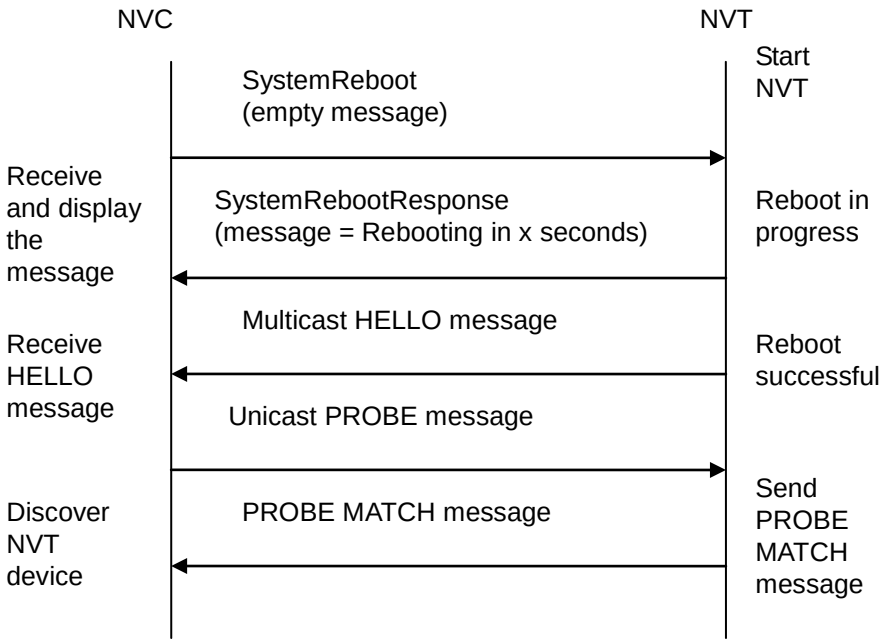
WSDL Reference: devicemgmt.wsdl

Test Purpose: To reboot the NVT through SystemReboot command.

Pre-Requisite: None.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SystemReboot message to reset the NVT.
4. Verify that NVT sends SystemRebootResponse message (example message string = “Rebooting in x seconds”).
5. NVT will send Multicast HELLO message after it is successfully rebooted.
6. NVC will verify the HELLO message sent by NVT.
7. NVC will send Unicast PROBE message to discover the NVT.
8. NVT will send a PROBE MATCH message.

9. NVC will verify the PROBE MATCH message sent by NVT.

Note: If BYE message is supported by NVT, then NVT shall send multicast BYE message before the reboot.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SystemRebootResponse message.

The DUT did not send HELLO message.

The DUT did not send PROBE MATCH message.

6.3.9 NVT SYSTEM COMMAND DEVICE INFORMATION

Test Label: Device Management NVT System Command GetDeviceInformation Test.

ONVIF Core Specification Coverage: 8.3.1 Device Information

Command Under Test: GetDeviceInformation

Device Type: NVT

Requirement Level: MUST

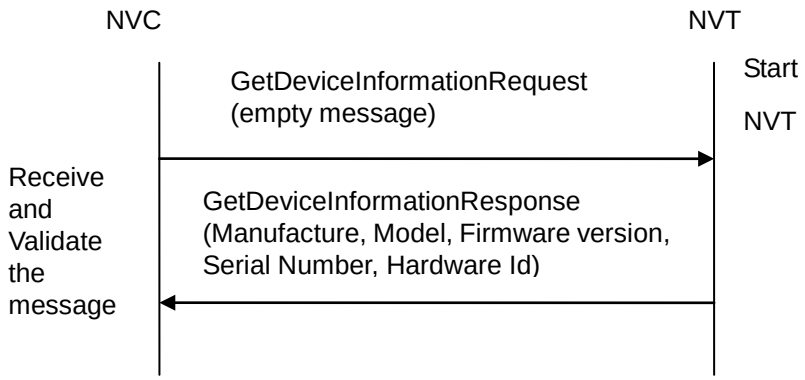
WSDL Reference: devicemgmt.wsdl

Test Purpose: To retrieve device information of NVT through GetDeviceInformation command.

Pre-Requisite: None.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetDeviceInformationRequest message to retrieve device information such as manufacture, model and firmware version etc.
4. Verify the GetDeviceInformationResponse from NVT (Manufacture, Model, Firmware version, Serial Number and Hardware Id).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetDeviceInformationResponse message.

The DUT did not send one or more mandatory information in the GetDeviceInformationResponse message (mandatory information - Manufacturer, Model, Firmware Version, Serial Number and Hardware Id)

6.4 Security

6.4.1 NVT SECURITY COMMAND GETUSERS

Test Label: Device Management NVT Security Command **GetUsers** Verification.

ONVIF Core Specification Coverage: 8.4.3 Get users.

Device Type: NVT.

Command Under Test: GetUsers.

Requirement Level: MUST.

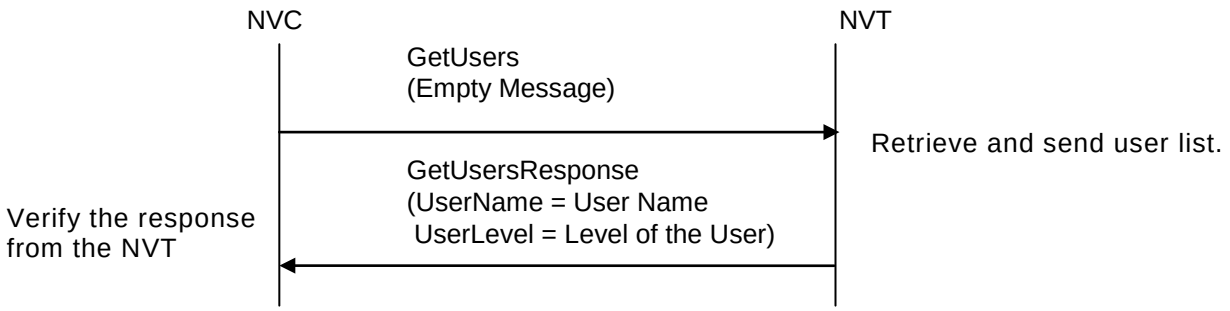
WSDL Reference: devicemgmt.wsdl

Test Purpose: To verify the behaviour of GetUsers command.

Pre-Requisite: It might be necessary that, NVC may need to operate in administrator mode to execute this test case.

Test Configuration: NVC and NVT.

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.
4. Verify that NVT sends the GetUsersResponse message (UserName, UserLevel).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetUsersResponse message.

Note: NVT may respond with none or more than one users.

6.4.2 NVT SECURITY COMMAND CREATEUSERS

Test Label: Device Management NVT Security Command **CreateUsers** Verification.

ONVIF Core Specification Coverage: 8.4.4 Create users.

Device Type: NVT.

Command Under Test: CreateUsers

Requirement Level: MUST.

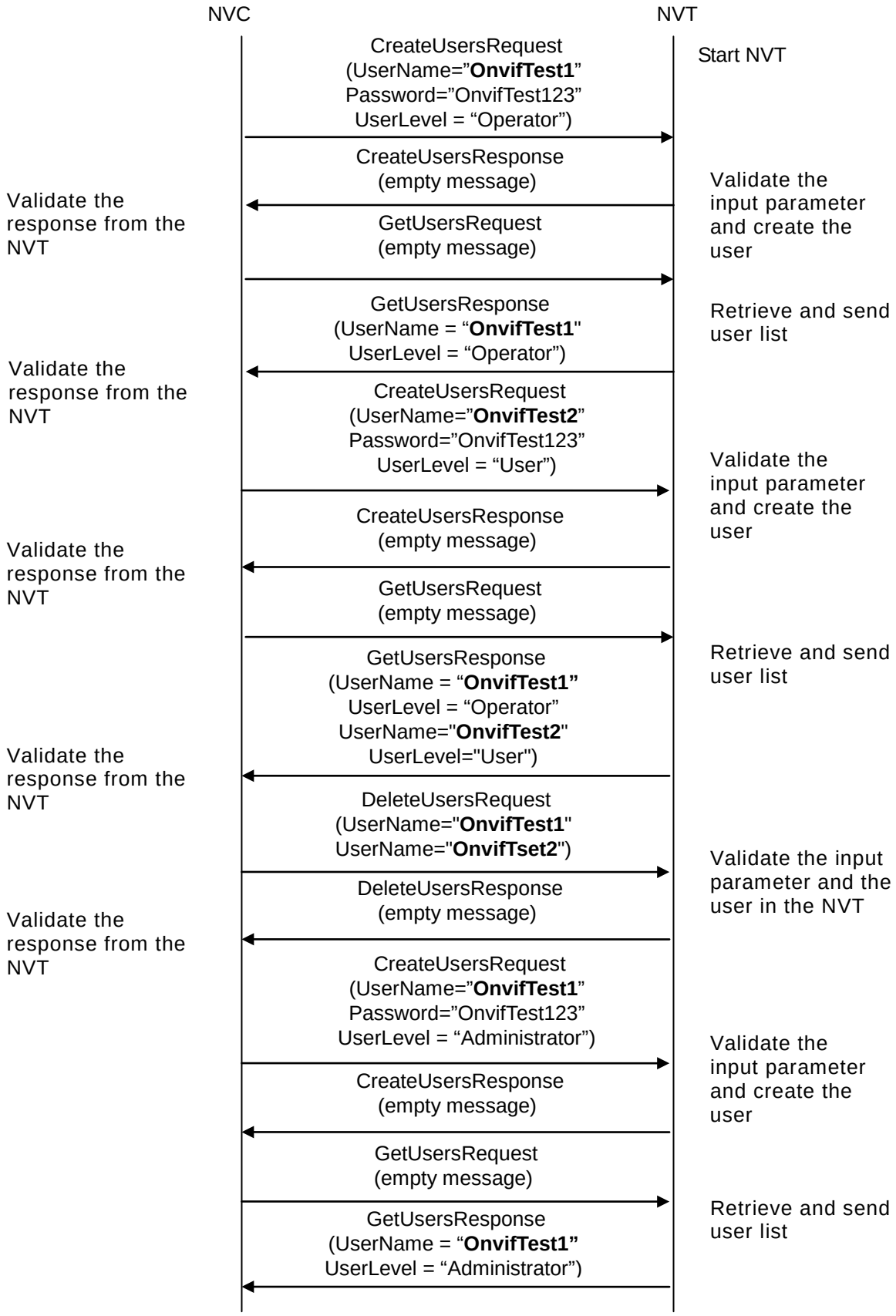
WSDL Reference: devicemgmt.wsdl

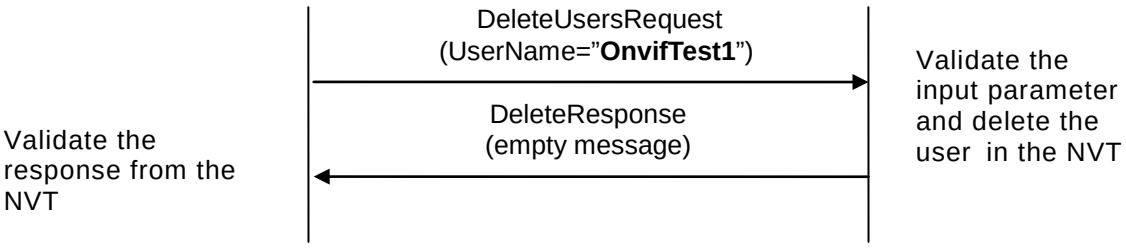
Test Purpose: To verify the behaviour of CreateUsers command.

Pre-Requisite: It might be necessary that, NVC may need to operate in administrator mode to execute this test case.

Test Configuration: NVC and NVT.

Test Sequence:


NVC
NVT
CreateUsersRequest
(UserName="OnvifTest1"
Password="OnvifTest123"
UserLevel = "Operator")
Start NVT
CreateUsersResponse
(empty message)
Validate the input parameter and create the user
Validate the response from the NVT
GetUsersRequest
(empty message)
GetUsersResponse
(UserName = "OnvifTest1"
UserLevel = "Operator")
Retrieve and send user list
Validate the response from the NVT
CreateUsersRequest
(UserName="OnvifTest2"
Password="OnvifTest123"
UserLevel = "User")
Validate the input parameter and create the user
CreateUsersResponse
(empty message)
Validate the response from the NVT
GetUsersRequest
(empty message)
GetUsersResponse
(UserName = "OnvifTest1"
UserLevel = "Operator"
UserName="OnvifTest2"
UserLevel="User")
Retrieve and send user list
Validate the response from the NVT
DeleteUsersRequest
(UserName="OnvifTest1"
UserName="OnvifTset2")
Validate the input parameter and the user in the NVT
DeleteUsersResponse
(empty message)
Validate the response from the NVT
CreateUsersRequest
(UserName="OnvifTest1"
Password="OnvifTest123"
UserLevel = "Administrator")
Validate the input parameter and create the user
CreateUsersResponse
(empty message)
GetUsersRequest
(empty message)
GetUsersResponse
(UserName = "OnvifTest1"
UserLevel = "Administrator")
Retrieve and send user list



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke CreateUsersRequest message (UserName="OnvifTest1" Password="OnvifTest123" UserLevel = "Operator"), to create the user in the NVT.
4. Verify that NVT sends CreateUsersResponse message (empty message).
5. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.
6. Verify that NVT sends GetUsersResponse message (UserName = "OnvifTest1", UserLevel = "Operator").
7. NVC will invoke CreateUsersRequest message (UserName = "OnvifTest2" Password = "OnvifTest123" UserLevel = "User") to create the user in the NVT.
8. Verify that NVT sends CreateUsersResponse message (empty message).
9. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.
10. Verify that NVT sends the GetUsersResponse message (UserName = "OnvifTest1" UserLevel = "Operator", UserName ="OnvifTest2" UserLevel = "User").
11. NVC will invoke DeleteUsersRequest message (UserName = "OnvifTest1", UserName = "OnvifTest2") to delete the users in the NVT.
12. Verify that NVT sends DeleteUsersResponse message (empty Message).
13. NVC will invoke CreateUsersRequest message (UserName="OnvifTest1" Password="OnvifTest123" UserLevel="Administrator"), to create the user in the NVT.
14. Verify that NVT sends CreateUsersResponse message (empty message).
15. NVC will invoke GetUsersRequest message (empty message) to retrieve the user list from the NVT.
16. Verify that NVT sends GetUsersResponse message (UserName="OnvifTest1", UserLevel="Administrator").
17. NVC will invoke DeleteUsersRequest message (UserName="OnvifTest1") to delete the users in the NVT.
18. Verify that NVT sends DeleteUsersResponse message (empty message).

Test Result:

PASS –

DUT passes all assertions

DUT creates user either at step 3 or step 10 or step 16 successfully or DUT creates users at step 3, step 10 and step 16 successfully .

FAIL –

The DUT did not send GetUsersResponse message.

The DUT did not send CreateUsersResponse message.

The DUT did not send DeleteUsersResponse message.

Note:

1 It may be possible that NVT may return SOAP Fault 1.2 "TooManyUsers" for the CreateUsers command. Such SOAP 1.2 fault message shall be treated as PASS case for this test.

2 It may be possible that NVT may return more number of users than actually created in this test case i.e. default users if any might be returned.

3 In some NVT's it might not be possible to create user with all levels. So if the NVT successfully creates the user either at step 3 or step 10 or step 16 successfully then this test case shall be treated as PASS case.

6.4.3 NVT SECURITY COMMAND CREATEUSERS ERROR CASE

Test Label: Device Management NVT Security Command **CreateUsers** Verification, Creating Duplicate Users.

ONVIF Core Specification Coverage: 8.4.4 Create Users.

Device Type: NVT.

Command Under Test: CreateUsers.

Requirement Level: SHOULD.

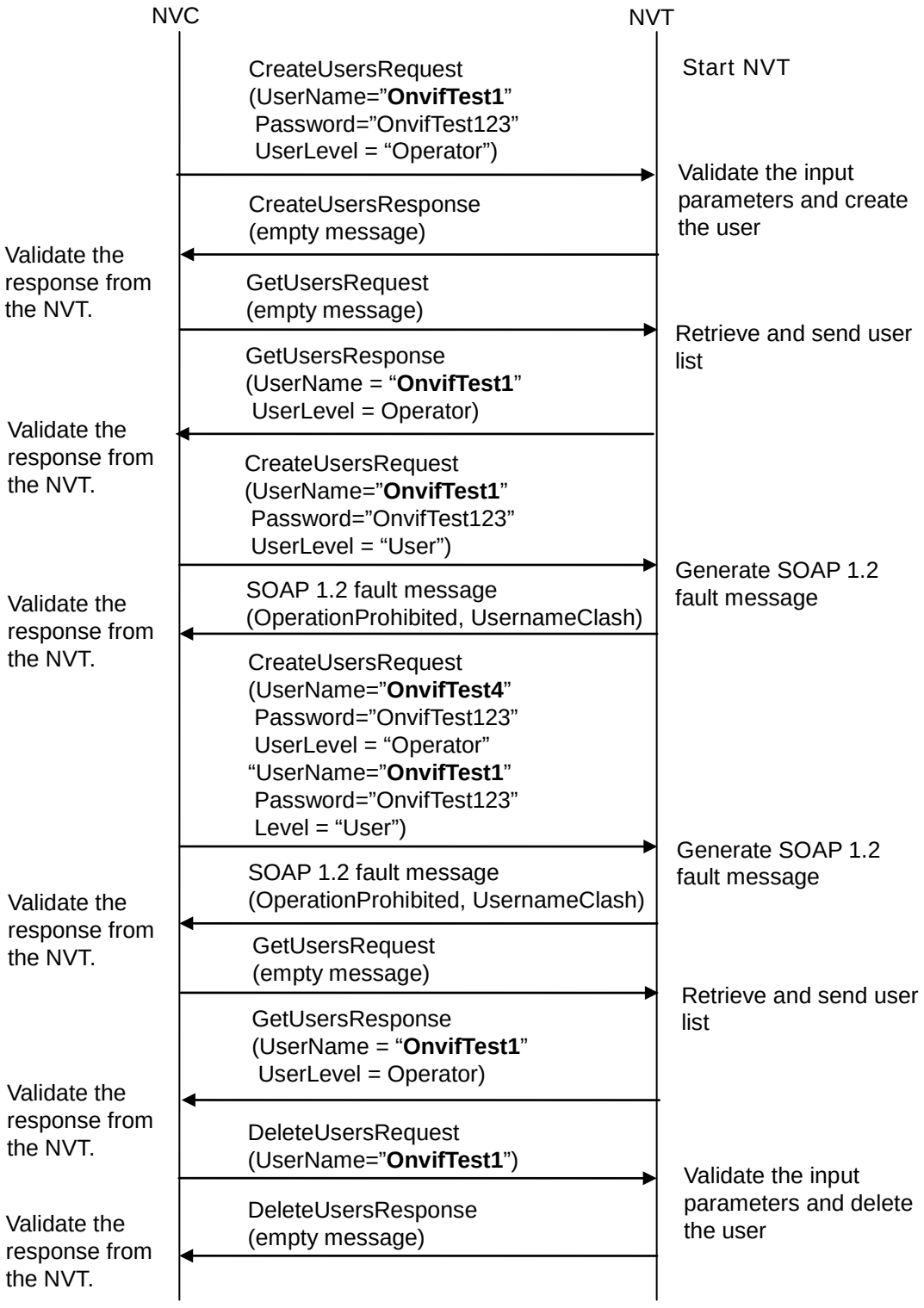
WSDL Reference: devicemgmt.wsdl

Test Purpose: To verify the behaviour of CreateUsers command, if the user is being created already exists.

Pre-Requisite: It might be necessary that, NVC may need to operate in administrator mode to execute this test case.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke CreateUsersRequest message (UserName = "OnvifTest1" Password="OnvifTest123" UserLevel = "Operator"), to create the user in the NVT.

4. Verify that NVT sends CreateUsersResponse message (empty message).
5. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.
6. Verify that NVT sends GetUsersResponse message (UserName = “OnvifTest1” UserLevel = Operator)
7. NVC will invoke CreateUsersRequest message (UserName = ”OnvifTest1” Password = OnvifTest123 UserLevel = User) to create the user in the NVT.
8. Verify that NVC generates SOAP 1.2 fault message (OperationProhibited, UsernameClash).
9. NVC will invoke CreateUsersRequest message (UserName = “OnvifTest4” Password = “OnvifTest123” UserLevel = “Operator”, UserName = “OnvifTest1” Password = “OnvifTest123” UserLevel = “User”) to create the users in the NVT.
10. Verify that NVC generates SOAP 1.2 fault message (OperationProhibited, UsernameClash).
11. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.
12. Verify that NVT sends GetUsersResponse message (UserName = “OnvifTest1” UserLevel = Operator)
13. NVC will invoke DeleteUsersRequest message (UserName = ”OnvifTest1”) to delete the user in the NVT.
14. Verify that NVT sends DeleteUsersResponse message (empty message).

Test Result:

PASS –

DUT passes all assertions

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct SOAP 1.2 fault message (OperationProhibited, UsernameClash).

The DUT did not send GetUsers response message.

The DUT did not send CreateUsers response message.

The DUT did not send DeleteUsers response message.

The DUT creates the user “OnvifTest1” with Level = “User”.

The DUT creates the user “OnvifTest4”.

Note:

1 It may be possible that NVT may return SOAP Fault 1.2 “TooManyUsers” for the CreateUsers command. Such SOAP 1.2 fault message shall be treated as PASS case for this test.

2 It may be possible that NVT may return more number of users than actually created in this test case i.e. default users if any might be returned.

6.4.4 NVT SECURITY COMMAND DELETEUSERS

Test Label: Device Management NVT Security Command **DeleteUsers** Verification.

ONVIF Core Specification Coverage: 8.4.5 Delete users.

Device Type: NVT.

Command Under Test: DeleteUsers

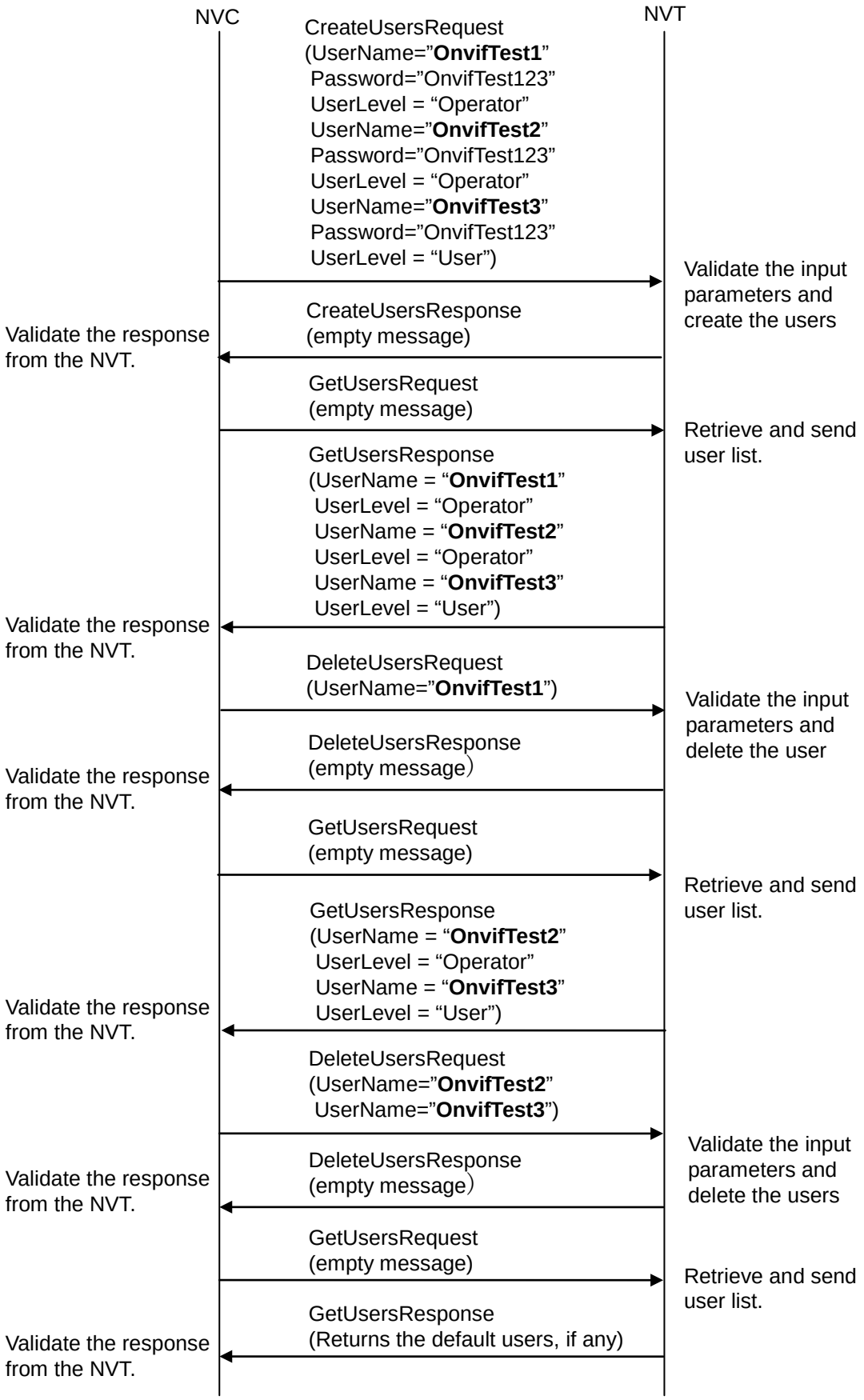
Requirement Level: MUST.

WSDL Reference: devicemgmt.wsdl

Pre-Requisite: It might be necessary that, NVC may need to operate in administrator mode to execute this test case.

Test Configuration: NVC and NVT.

Test Sequence:


NVC
NVT
CreateUsersRequest
(UserName="OnvifTest1"
Password="OnvifTest123"
UserLevel = "Operator"
UserName="OnvifTest2"
Password="OnvifTest123"
UserLevel = "Operator"
UserName="OnvifTest3"
Password="OnvifTest123"
UserLevel = "User")
Validate the input parameters and create the users
CreateUsersResponse
(empty message)
Validate the response from the NVT.
GetUsersRequest
(empty message)
Retrieve and send user list.
GetUsersResponse
(UserName = "OnvifTest1"
UserLevel = "Operator"
UserName = "OnvifTest2"
UserLevel = "Operator"
UserName = "OnvifTest3"
UserLevel = "User")
Validate the response from the NVT.
DeleteUsersRequest
(UserName="OnvifTest1")
Validate the input parameters and delete the user
DeleteUsersResponse
(empty message)
Validate the response from the NVT.
GetUsersRequest
(empty message)
Retrieve and send user list.
GetUsersResponse
(UserName = "OnvifTest2"
UserLevel = "Operator"
UserName = "OnvifTest3"
UserLevel = "User")
Validate the response from the NVT.
DeleteUsersRequest
(UserName="OnvifTest2"
UserName="OnvifTest3")
Validate the input parameters and delete the users
DeleteUsersResponse
(empty message)
Validate the response from the NVT.
GetUsersRequest
(empty message)
Retrieve and send user list.
GetUsersResponse
(Returns the default users, if any)
Validate the response from the NVT.

Test Procedure:

1. Start an NVC
2. Start an NVT.
3. NVC will invoke CreateUsersRequest message (UserName= ”OnvifTest1” Password= ”OnvifTest123” UserLevel = “Operator”, UserName= ”OnvifTest2” Password= ”OnvifTest123” UserLevel = “Operator”, UserName= ”OnvifTest3” Password= ”OnvifTest123” UserLevel = “User”) to create the user in the NVT.
4. Verify that NVT sends CreateUsersResponse message (empty message).
5. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.
6. Verify that NVT sends GetUsersResponse message (UserName = “OnvifTest1” UserLevel = “Operator”, UserName = “OnvifTest2” UserLevel = “Operator”, UserName = “OnvifTest3” UserLevel = “User”).
7. NVC will invoke DeleteUsersRequest message (UserName=”OnvifTest1”), to delete the user.
8. Verify that NVT sends DeleteUsersResponse message (empty message).
9. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.
10. Verify that NVT sends the updated user list (UserName = “OnvifTest2” UserLevel = “Operator”, UserName = “OnvifTest3” UserLevel = “User”).
11. NVC will invoke DeleteUsersRequest message (UserName = “OnvifTest2”, UserName = “OnvifTest3”) to delete the users.
12. Verify that NVT sends the DeleteUserResponse message (empty message).
13. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.
14. Verify that NVT sends the GetUsersResponse message (Returns the default users, if any).

Test Result:

PASS –

DUT passes all assertions

FAIL –

The DUT did not send DeleteUsers response message.

The DUT did not create the users at step 3.

The DUT did not send GetUsers response message.

The DUT did not send CreateUsers response message

The DUT did not delete the users.

Note:

1 It may be possible that NVT may return SOAP Fault 1.2 “TooManyUsers” for the CreateUsers command. Such SOAP 1.2 fault message shall be treated as PASS case for this test.

2 It may be possible that NVT may return more number of users than actually created in this test case i.e. default users if any might be returned.

6.4.5 NVT SECURITY COMMAND DELETEUSERS ERROR CASE

Test Label: Device Management NVT Security Command DeleteUsers Verification, Deleting Non Existing Users.

ONVIF Core Specification Coverage: 8.4.5 Delete users.

Device Type: NVT.

Command Under Test: DeleteUsers.

Requirement Level: SHOULD.

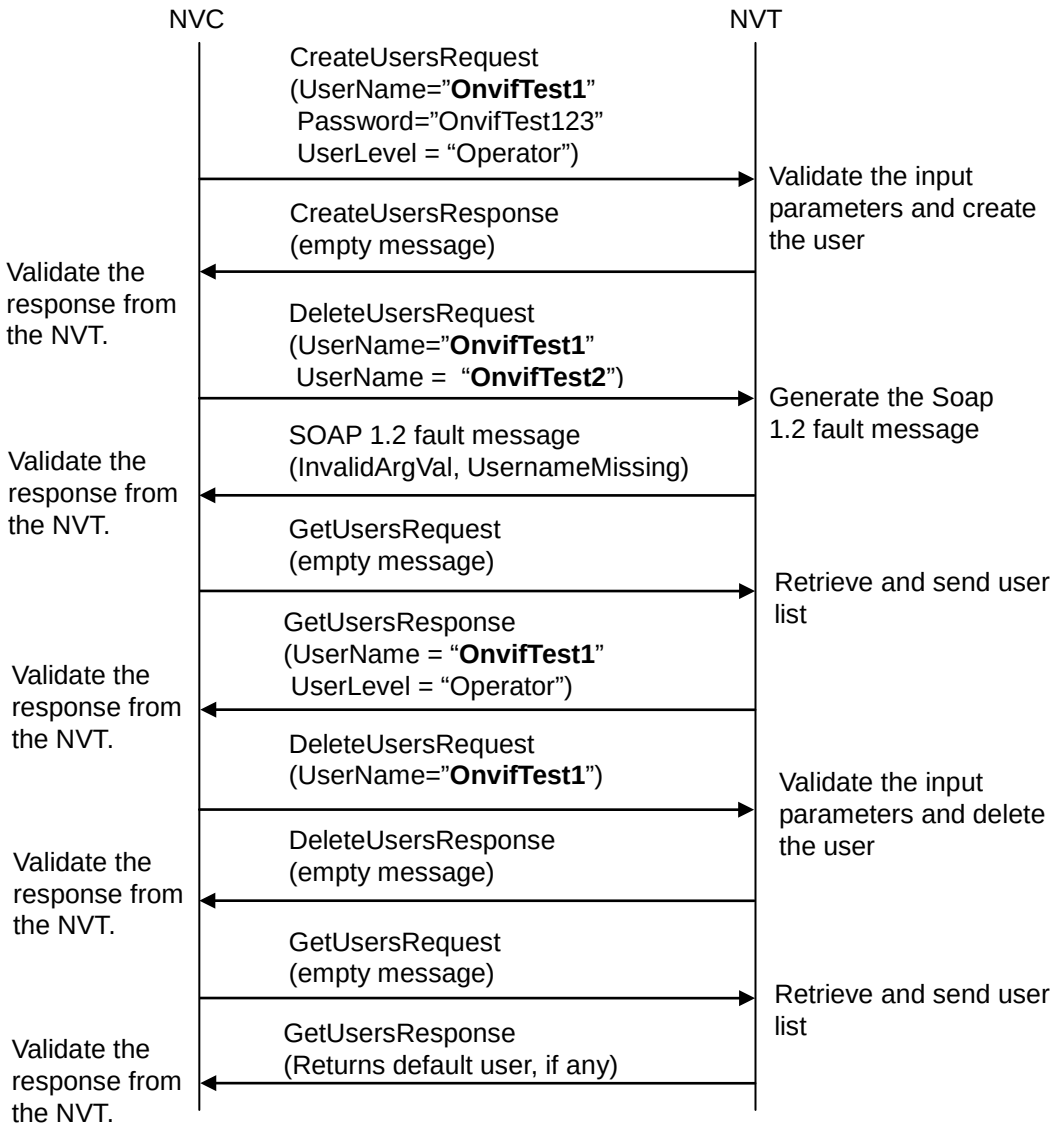
WSDL Reference: devicemgmt.wsdl

Test Purpose: To verify the behaviour of the DeleteUsers command, if the non-existing user is deleted.

Pre-Requisite: It might be necessary that, NVC may need to operate in administrator mode to execute this test case.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke CreateUsersRequest message (UserName = "OnvifTest1" Password =" OnvifTest123" UserLevel = "Operator"), to create the user in the NVT.
4. Verify that NVT sends CreateUsersResponse message (empty message).
5. NVC will invoke DeleteUsersRequest message (UserName = "OnvifTest1", UserName = "OnvifTest2"), to delete user.
6. Verify that the NVT responds with Soap 1.2 fault message (InvalidArgVal, UsernameMissing).
7. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.

8. Verify that NVT sends GetUsersResponse message (UserName = “OnvifTest1” UserLevel = Operator).
9. NVC will invoke DeleteUsersRequest message (UserName = “OnvifTest1”), to delete the user.
10. Verify that NVT sends DeleteUsersResponse message (empty message).
11. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.
12. Verify that NVT sends GetUsersResponse message (Returns default users, if any).

Test Result:

PASS –

DUT passes all assertions

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct SOAP 1.2 fault message (InvalidArgVal, UsernameMissing).

The DUT deletes the user “OnvifTest1” at step 5.

The DUT did not send DeleteUsersResponse message.

The DUT did not send CreateUsersResponse message.

The DUT did not send GetUsersResponse message.

Note:

1 It may be possible that NVT may return SOAP Fault 1.2 “TooManyUsers” for the CreateUsers command. Such SOAP 1.2 fault message shall be treated as PASS case for this test.

2 It may be possible that NVT may return more number of users than actually created in this test case i.e. default users if any might be returned.

6.4.6 NVT SECURITY COMMAND DELETEUSERS DELETE ALL USERS

Test Label: Device Management NVT Security Command DeleteUsers Verification, Deleting All Users.

ONVIF Core Specification Coverage: 8.4.5 Delete users.

Device Type: NVT.

Command Under Test: DeleteUsers.

Requirement Level: MUST.

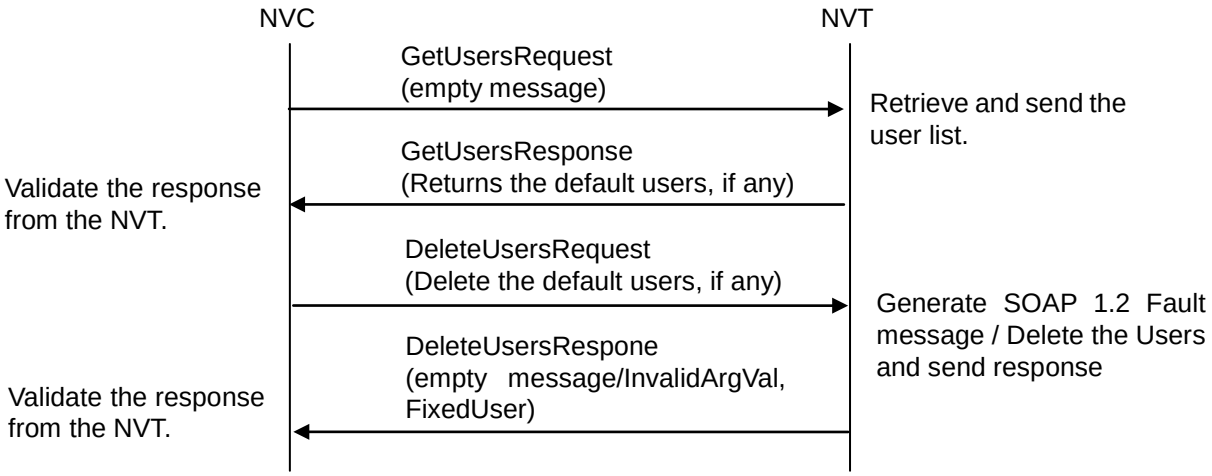
WSDL Reference: devicemgmt.wsdl

Test Purpose: To verify the behaviour of the DeleteUsers command, when all the users are deleted.

Pre-Requisite: It might be necessary that, NVC may need to operate in administrator mode to execute this test case.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.
4. Verify that NVT sends GetUsersResponse message (Returns default users, if any).
5. NVC will invoke DeleteUsersRequest message (UserName= Default Users), to delete the default users if there are any.
6. Verify that NVT sends DeleteUsersResponse message (empty message / InvalidArgVal, FixedUser). Depending upon the implementation NVT may respond with SOAP 1.2 fault message or send empty response.

PASS –

DUT passes all assertions

FAIL –

The DUT did not send correct SOAP 1.2 fault message (InvalidArgVal, FixedUser).

The DUT did not send GetUsersResponse message.

The DUT did not send DeleteUserResponse message.

Note:

1 NVT may return the default users, if any.

2 It is not be possible to recover the default user if is deleted during the test case execution.

3 The original user, used to access the NVT, must be restored in case all users were deleted.

6.4.7 NVT SECURITY COMMAND SETUSER

Test Label: Device Management NVT Security Command **SetUser** Verification.

ONVIF Core Specification Coverage: 8.4.6 Set users.

Device Type: NVT.

Command Under Test: SetUser.

Requirement Level: MUST.

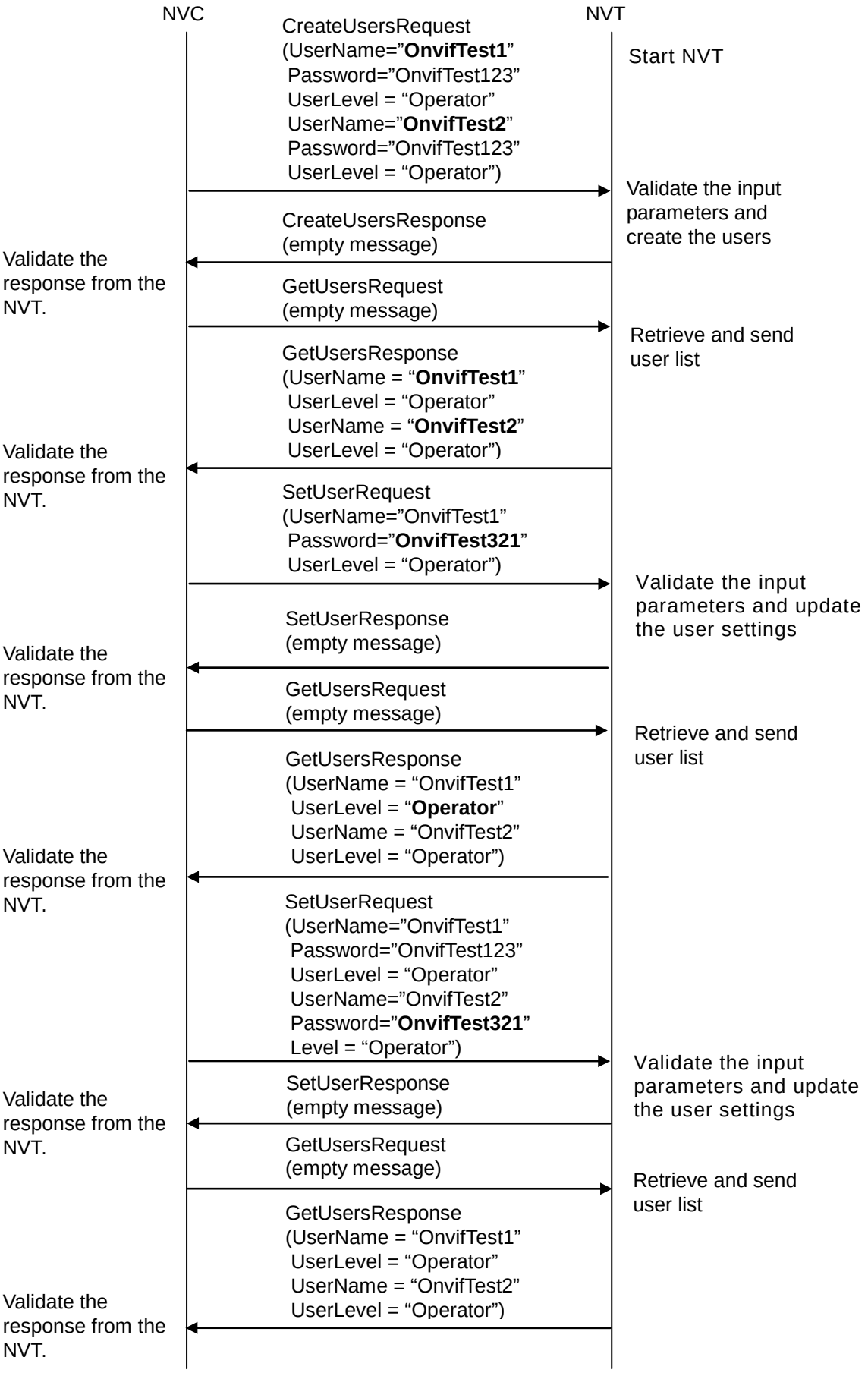
WSDL Reference: devicemgmt.wsdl

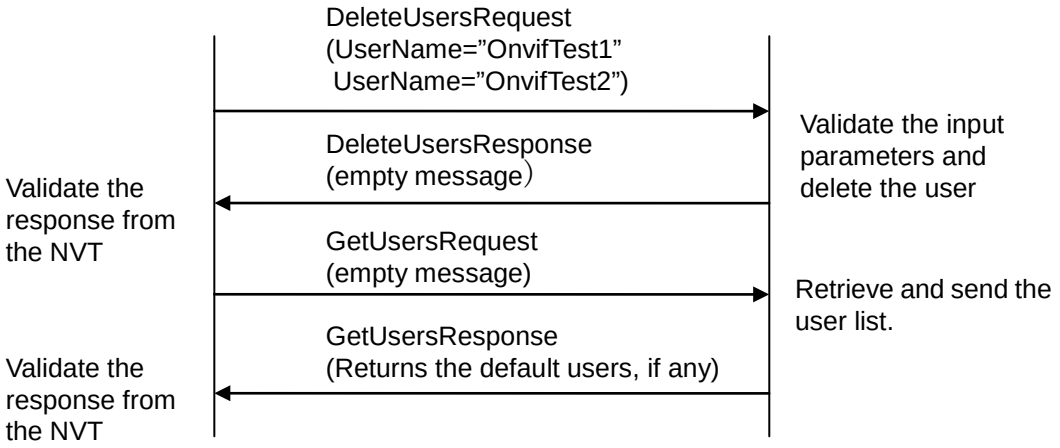
Test Purpose: To verify the behaviour of SetUser command.

Pre-Requisite: It might be necessary that, NVC may need to operate in administrator mode to execute this test case.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
CreateUsersRequest
(UserName="OnvifTest1"
Password="OnvifTest123"
UserLevel = "Operator"
UserName="OnvifTest2"
Password="OnvifTest123"
UserLevel = "Operator")
Start NVT
Validate the input parameters and create the users
CreateUsersResponse
(empty message)
Validate the response from the NVT.
GetUsersRequest
(empty message)
Retrieve and send user list
GetUsersResponse
(UserName = "OnvifTest1"
UserLevel = "Operator"
UserName = "OnvifTest2"
UserLevel = "Operator")
Validate the response from the NVT.
SetUserRequest
(UserName="OnvifTest1"
Password="OnvifTest321"
UserLevel = "Operator")
Validate the input parameters and update the user settings
SetUserResponse
(empty message)
Validate the response from the NVT.
GetUsersRequest
(empty message)
Retrieve and send user list
GetUsersResponse
(UserName = "OnvifTest1"
UserLevel = "Operator"
UserName = "OnvifTest2"
UserLevel = "Operator")
Validate the response from the NVT.
SetUserRequest
(UserName="OnvifTest1"
Password="OnvifTest123"
UserLevel = "Operator"
UserName="OnvifTest2"
Password="OnvifTest321"
Level = "Operator")
Validate the input parameters and update the user settings
SetUserResponse
(empty message)
Validate the response from the NVT.
GetUsersRequest
(empty message)
Retrieve and send user list
GetUsersResponse
(UserName = "OnvifTest1"
UserLevel = "Operator"
UserName = "OnvifTest2"
UserLevel = "Operator")
Validate the response from the NVT.



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke CreateUsersRequest message (UserName = "OnvifTest1" Password = "OnvifTest123" UserLevel = "Operator", UserName = "OnvifTest2" Password = "OnvifTest123" UserLevel = "Operator"), to create the user in the NVT.
4. Verify that NVT sends CreateUsersResponse message (empty message).
5. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.
6. Verify that NVT sends GetUsersResponse message (UserName = "OnvifTest1" UserLevel = Operator, UserName = "OnvifTest2" UserLevel = "Operator").
7. NVC will invoke SetUserRequest message (UserName = "OnvifTest1" Password = "OnvifTest321" UserLevel = "Operator"), to update the user settings in the NVT.
8. Verify that NVT sends SetUserResponse message (empty message).
9. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.
10. Verify that NVT sends GetUsersResponse message (UserName= "OnvifTest1" UserLevel = "Operator", UserName = "OnvifTest2" UserLevel = "Operator").
11. NVC will invoke SetUserRequest message (UserName = "OnvifTest1" Password = "OnvifTest123" UserLevel = "Operator", UserName = "OnvifTest2" Password = "OnvifTest321" UserLevel = "Operator"), to update user settings in the NVT.
12. Verify that NVT sends SetUserResponse message (empty message).
13. NVC will invoke GetUsersRequest (empty message), to retrieve the user list from the NVT.
14. Verify that NVT sends GetUsersResponse message (UserName = OnvifTest1 UserLevel = "Operator", UserName = OnvifTest2 UserLevel = "User").
15. NVC will invoke DeleteUsersRequest message (UserName = "OnvifTest1", UserName = "OnvifTest2") to delete the user in the NVT.
16. Verify that NVT sends DeleteUsersResponse message (Empty Message).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not update the settings of the user(s).

The DUT did not send GetUsersResponse message.

The DUT did not send SetUserResponse message.

The DUT did not send DeleteUsersResponse message.

The DUT did not send CreateUsersResponse message.

Note:

1 It may be possible that NVT may return SOAP Fault 1.2 “TooManyUsers” for the CreateUsers command. In this case, test must be executed with the default users if any.

2 It may be possible that NVT may return more number of users than actually created in this test case, i.e. default users if any might be returned.

6.4.8 NVT SECURITY COMMAND USER MANAGEMENT ERROR CASE

Test Label: Device Management NVT Security Command **SetUser** Verification, User Settings Non Existing User.

ONVIF Core Specification Coverage: 8.4.6 Set users.

Device Type: NVT

Command Under Test: SetUser.

Requirement Level: SHOULD

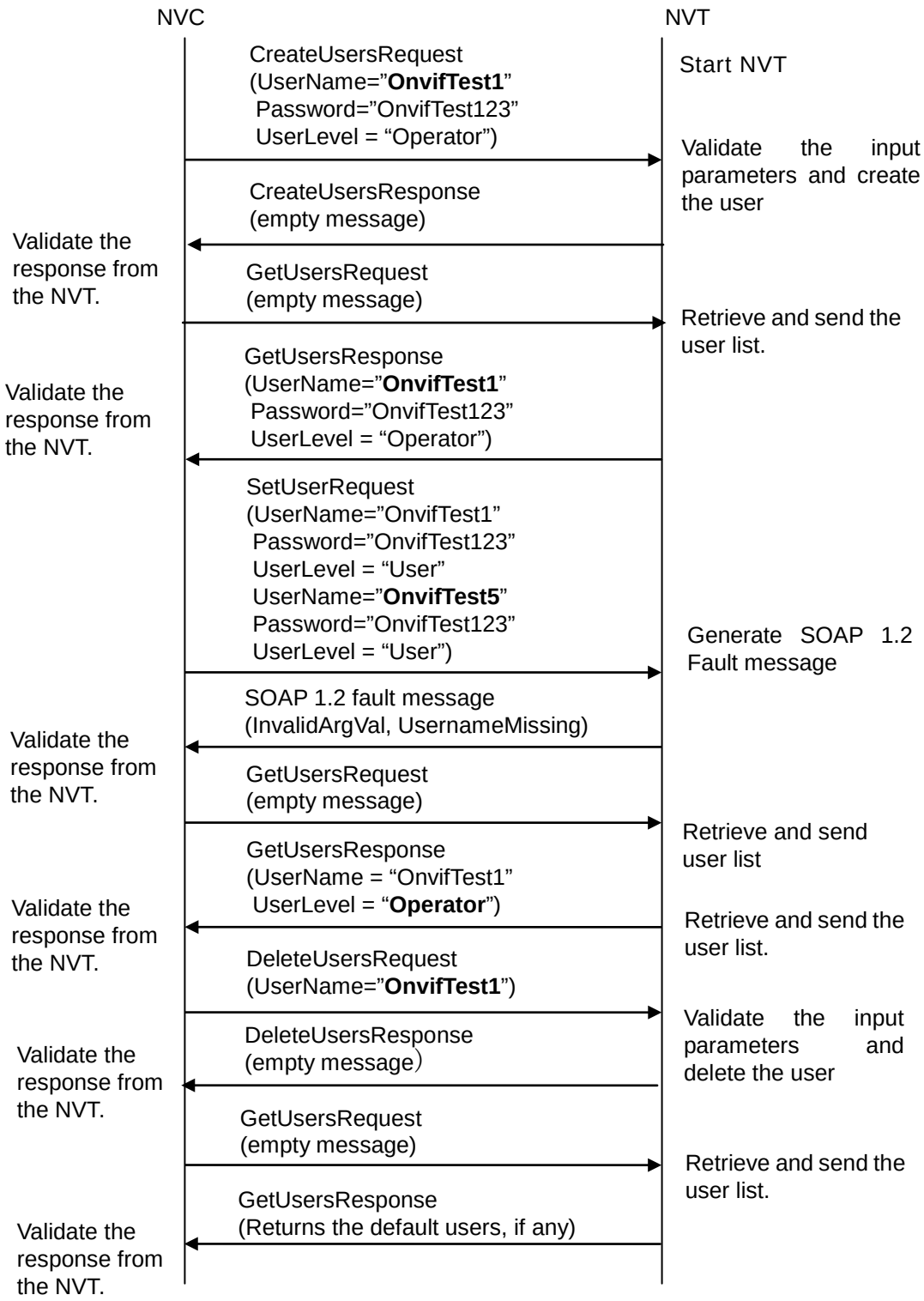
WSDL Reference: devicemgmt.wsdl

Test Purpose: To verify the behaviour of the SetUser command when updating the settings of non-existing user.

Pre-Requisite: It might be necessary that, NVC may need to operate in administrator mode to execute this test case.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.

2. Start an NVT.
3. NVC will invoke CreateUsersRequest message (UserName = ”OnvifTest1” Password = ”OnvifTest123” UserLevel = “Operator”), to create the user in the NVT.
4. Verify that NVT sends CreateUsersResponse message (empty message).
5. NVC will invoke SetUserRequest message (UserName = “OnvifTest1” Password=OnvifTest123 UserLevel = “User”, UserName = “OnvifTest5” Password = “OnvifTest123” UserLevel = “User”) to update the user settings in the NVT.
6. Verify that NVT generates SOAP 1.2 fault message (InvalidArgVal, UsernameMissing).
7. NVC will invoke GetUsersRequest message (empty message), to retrieve the user list from the NVT.
8. Verify that NVT sends the GetUsersResponse message (UserName = “OnvifTest1” UserLevel = “Operator”).
9. NVC will invoke DeleteUsersRequest message (UserName = ”OnvifTest1”), to delete the user in the NVT.
10. Verify that NVT sends DeleteUsersResponse message (Empty Message).

Test Result:

PASS –

DUT passes all assertions

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct SOAP 1.2 fault message (InvalidArgVal, UsernameMissing).

The DUT did not send GetUsersResponse message.

The DUT did not send SetUserResponse message.

The DUT did not send CreateUsersResponse message.

The DUT did not send DeleteUsersResponse message.

The DUT updates the settings of the user “OnvifTest1”.

Note:

1 It may be possible that NVT may return SOAP Fault 1.2 “TooManyUsers” for the CreateUsers command. In this case, test must be executed with the default users if any.

2 It may be possible that NVT may return more number of users than actually created in this test case, i.e. default users if any might be returned.

7 Media Configuration Test Cases

7.1 Media Profile

7.1.1 NVT MEDIA PROFILE CONFIGURATION

Test Label: Media Configuration NVT Media Profiles.

ONVIF Core Specification Coverage: 10.2.2 Get media profiles

Device Type: NVT

Command Under Test: GetProfiles

WSDL Reference: media.wsdl

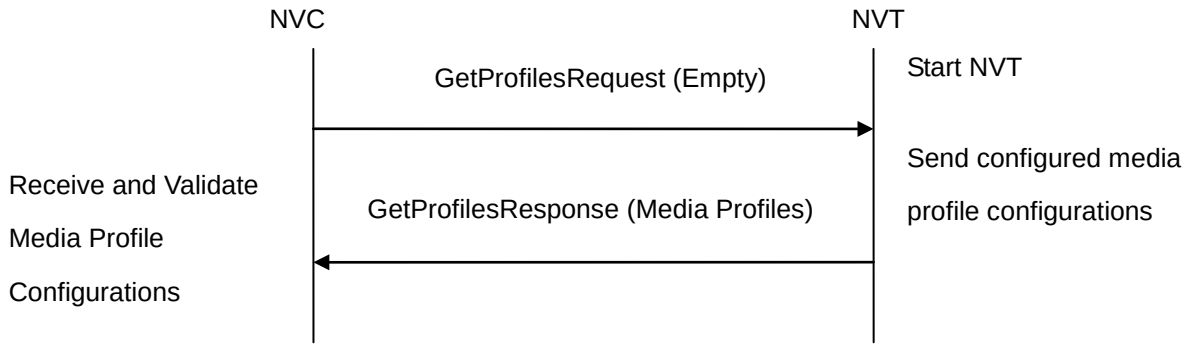
Requirement Level: MUST

Test Purpose: To retrieve existing media profiles of NVT and the corresponding media entities (video source and video encoder).

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetProfilesRequest message to retrieve existing media profiles of the NVT.
4. Verify that the NVT returns at least one media profile with video configuration (video source and video encoder) in the GetProfilesResponse message.
5. Verify that all media profile elements have ‘fixed’ attribute.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send GetProfilesResponse message.

DUT has no default media profile configuration.

DUT did not provide at least one media profile with video source and video encoder configurations.

One or more media profiles don't have 'fixed' attribute.

7.1.2 NVT DYNAMIC MEDIA PROFILE CONFIGURATION

Test Label: Media Configuration NVT Dynamic Media Profile Configuration.

ONVIF Core Specification Coverage: 10.2.1 Create media profile, 10.2.2 Get media profiles , 10.2.3 Get media profile, 10.2.4 Add video source configuration to a profile, 10.2.5 Add video encoder configuration to a profile, 10.2.11 Remove video source configuration from a profile, 10.2.12 Remove video encoder configuration from a profile, 10.2.18 Delete media profile.

Device Type: NVT

Command Under Test: None

WSDL Reference: media.wsdl

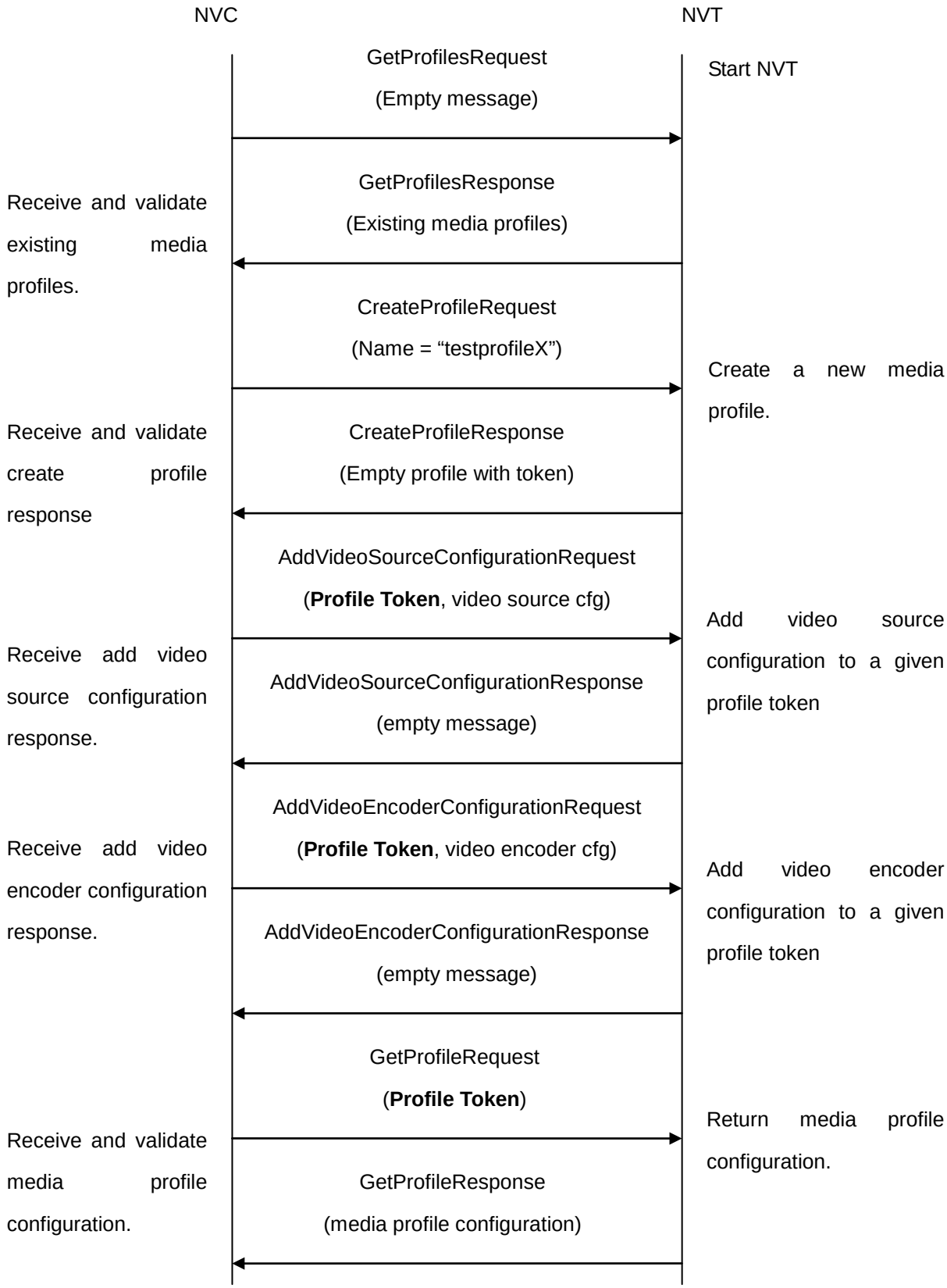
Requirement Level: MUST

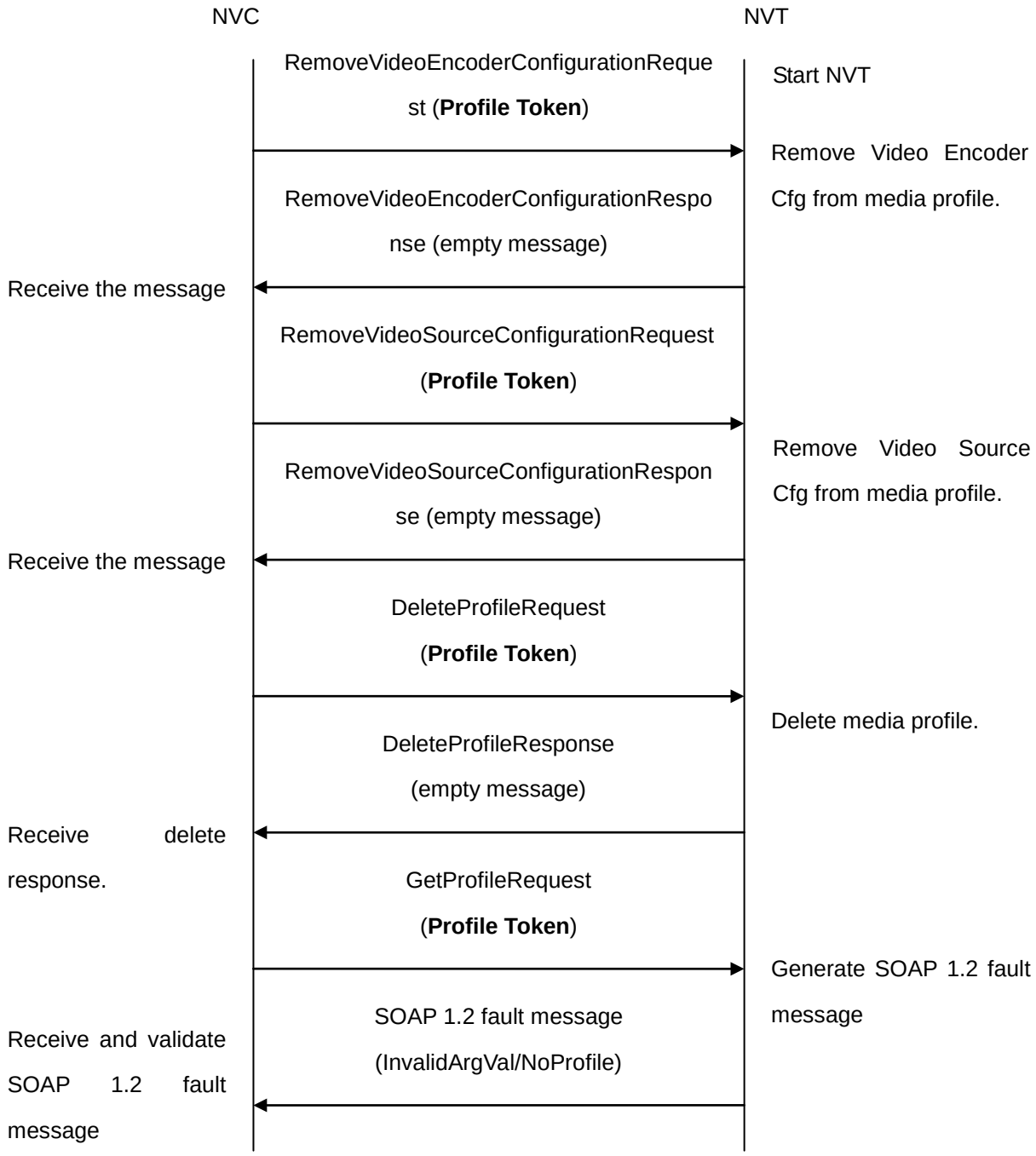
Test Purpose: To verify the behaviour of NVT for dynamic media profile configuration.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
GetProfilesRequest
(Empty message)
Start NVT
GetProfilesResponse
(Existing media profiles)
Receive and validate existing media profiles.
CreateProfileRequest
(Name = “testprofileX”)
Create a new media profile.
CreateProfileResponse
(Empty profile with token)
Receive and validate create profile response
AddVideoSourceConfigurationRequest
(Profile Token, video source cfg)
Add video source configuration to a given profile token
AddVideoSourceConfigurationResponse
(empty message)
Receive add video source configuration response.
AddVideoEncoderConfigurationRequest
(Profile Token, video encoder cfg)
Add video encoder configuration to a given profile token
AddVideoEncoderConfigurationResponse
(empty message)
Receive add video encoder configuration response.
GetProfileRequest
(Profile Token)
Return media profile configuration.
GetProfileResponse
(media profile configuration)
Receive and validate media profile configuration.



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetProfilesRequest message to retrieve existing media profiles configurations of the NVT.

4. Verify that the NVT returns at-least one media profile with video configuration (video source and video encoder) in GetProfilesResponse message.
5. NVC will invoke CreateProfileRequest message to create a new empty media profile.
6. NVT returns an empty profile with no profile entities in the CreateProfileResponse message.
7. NVC will invoke AddVideoSourceConfigurationRequest message (**Profile Token**, Reference to **Video Source Configuration of existing media profile**) to add video source configuration to new profile.
8. NVT sends AddVideoSourceConfigurationResponse message indicating successful addition of video source configuration.
9. NVC will invoke AddVideoEncoderConfigurationRequest message (**Profile Token**, Reference to **Video Encoder Configuration of existing media profile**) to add video encoder configuration to new profile.
10. NVT sends AddVideoEncoderConfigurationResponse message indicating successful addition of video encoder configuration.
11. NVC will invoke GetProfileRequest (**Profile Token**) message to verify video source and encoder configurations in a new profile.
12. NVT will return media profile configuration for requested media profile in the GetProfileResponse message.
13. NVC will invoke RemoveVideoEncoderConfigurationRequest message (**Profile Token**) to remove video encoder configuration from a media profile.
14. NVT sends RemoveVideoEncoderConfigurationResponse message indicating successfully removal of video encoder configuration.
15. NVC will invoke RemoveVideoSourceConfigurationRequest message (**Profile Token**) to remove video source configuration from a media profile.
16. NVT sends RemoveVideoSourceConfigurationResponse message indicating successfully removal of video source configuration.
17. NVC will invoke DeleteProfileRequest (**Profile Token**) message to delete the newly created media profile.
18. NVT will delete the media profile and sends DeleteProfileResponse message.
19. NVC will invoke GetProfileRequest (**deleted Profile Token**) message to check the existence of deleted media profile.
20. NVT will generate SOAP 1.2 fault message **(InvalidArgVal/NoProfile)**.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetProfilesResponse message.

The DUT did not send CreateProfileResponse message.

The DUT did not send AddVideoSourceConfigurationResponse message.

The DUT did not send AddVideoEncoderConfigurationResponse message.

The DUT did not send GetProfileResponse message.

The DUT did not set 'fixed' attribute of created media profile to 'false'

The DUT did not send RemoveVideoEncoderConfigurationResponse message.

The DUT did not send RemoveVideoSourceConfigurationResponse message.

The DUT did not send DeleteProfileResponse message.

The DUT did not send SOAP 1.2 fault message (InvalidArgVal/NoProfile).

Note: See Annex A.6 for Invalid SOAP 1.2 fault message definition.

7.2 Video Configuration

7.2.1 NVT VIDEO SOURCE CONFIGURATION

Test Label: Media Configuration NVT Video Source Configuration

ONVIF Core Specification Coverage: 10.2.2 Get media profiles, 10.3.1 GetVideoSources, 10.4.1 Get video source configurations, 10.4.2 Get video source configuration, 10.4.3 Get compatible video source configurations, 10.4.4 Get video source configuration options, 10.4.5 Modify a video source configuration

Device Type: NVT

Command Under Test: None

WSDL Reference: media.wsdl

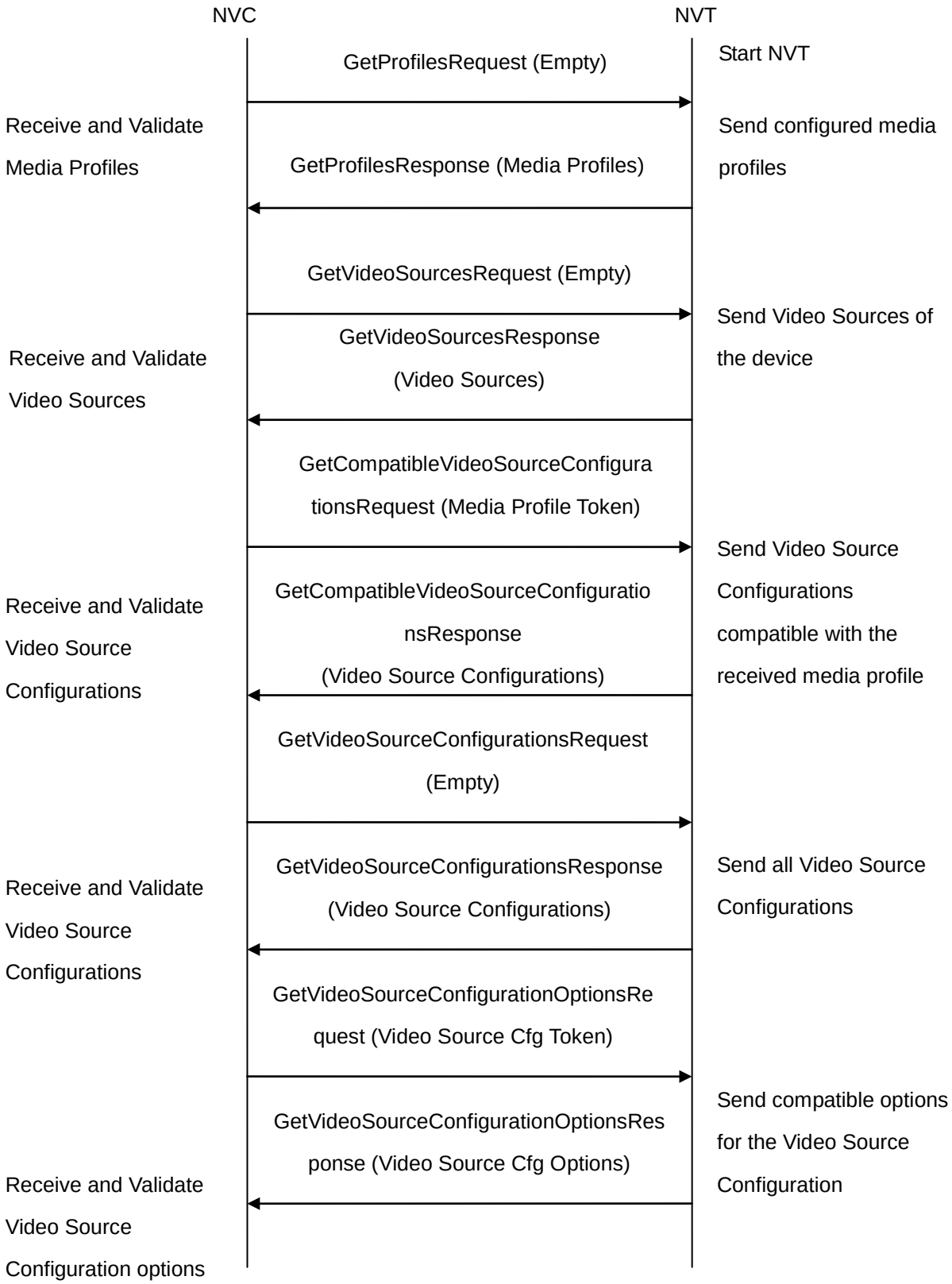
Requirement Level: MUST

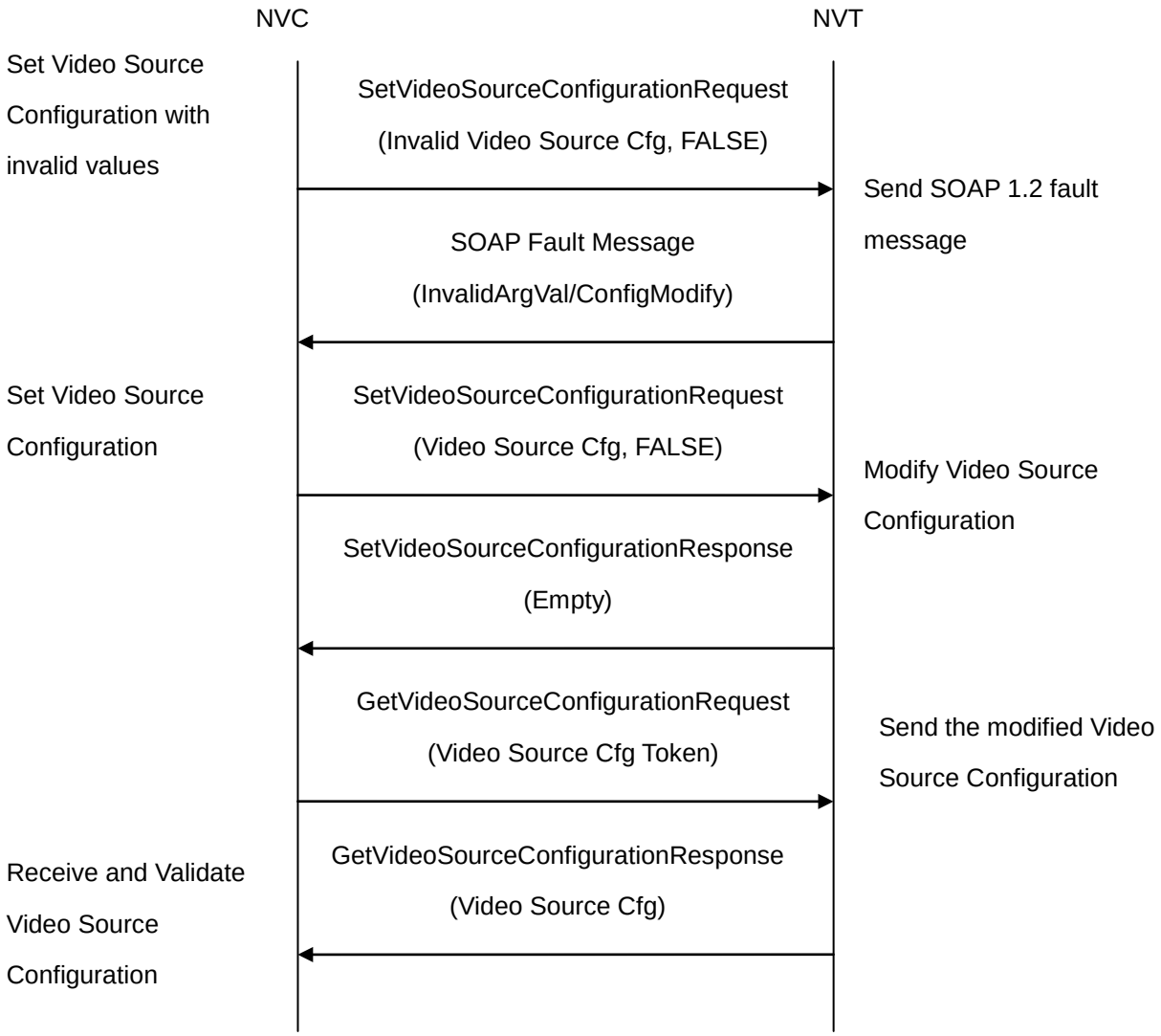
Test Purpose: To verify NVT Video Source Configuration Operations

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
GetProfilesRequest (Empty)
Start NVT
Receive and Validate Media Profiles
GetProfilesResponse (Media Profiles)
Send configured media profiles
GetVideoSourcesRequest (Empty)
Send Video Sources of the device
Receive and Validate Video Sources
GetVideoSourcesResponse (Video Sources)
GetCompatibleVideoSourceConfigurationsRequest (Media Profile Token)
Send Video Source Configurations compatible with the received media profile
Receive and Validate Video Source Configurations
GetCompatibleVideoSourceConfigurationsResponse (Video Source Configurations)
GetVideoSourceConfigurationsRequest (Empty)
Send all Video Source Configurations
Receive and Validate Video Source Configurations
GetVideoSourceConfigurationsResponse (Video Source Configurations)
GetVideoSourceConfigurationOptionsRequest (Video Source Cfg Token)
Send compatible options for the Video Source Configuration
Receive and Validate Video Source Configuration options
GetVideoSourceConfigurationOptionsResponse (Video Source Cfg Options)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes GetProfiles Request.
4. NVT sends the list of existing media profiles in GetProfiles Response message.
5. NVC invokes GetVideoSources request to retrieve the existing video sources of NVT.
6. NVC verifies the list of video sources sent by NVT.
7. NVC invokes GetCompatibleVideoSourceConfigurations request with one of the received media profile tokens as input argument.

8. NVT sends the list of video source configurations compatible with the received media profile token.
9. NVC verifies the list of video source configurations sent by NVT.
10. NVC invokes GetVideoSourceConfigurations request to retrieve the list of video source configurations supported by the NVT.
11. NVC verifies the list of video source configurations sent by the NVT.
12. NVC invokes GetVideoSourceConfigurationOptions request with one of the received video source configuration tokens as input argument.
13. NVT sends the range of configurable values supported by it for the received video source configuration token.
14. NVC invokes SetVideoSourceConfiguration request with video source configuration values outside the range specified in the GetVideoSourceConfigurationOptionsResponse and **'ForcePersistence'** flag as **'FALSE'**.
15. NVT send the SOAP 1.2 fault message **(InvalidArgVal/ConfigModify)**.
16. NVC verifies the SOAP fault message sent by NVT.
17. NVC invokes SetVideoSourceConfiguration request with video source configuration values within the range specified in GetVideoSourceConfigurationOptionsResponse and **'ForcePersistence'** flag as **'FALSE'**.
18. NVT modifies the video source configuration and sends the SetVideoSourceConfiguration response indicating success.
19. NVC invokes GetVideoSourceConfiguration request to verify the modified video source configuration.
20. NVT sends the modified video source configuration in GetVideoSourceConfiguration response.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send GetProfilesResponse message.

DUT did not send valid GetVideoSourcesResponse message.

DUT did not send valid GetCompatibleVideoSourceConfigurationsResponse message.

DUT did not send valid GetVideoSourceConfigurationsResponse message.

DUT did not send GetVideoSourceConfigurationOptionsResponse message.

DUT did not send the SOAP 1.2 fault message (InvalidArgVal/ConfigModify) for invalid SetVideoSourceConfiguration request.

DUT did not send SetVideoSourceConfigurationResponse message.

DUT did not send GetVideoSourceConfigurationResponse message.

DUT did not modify video source configuration correctly.

7.2.2 NVT VIDEO ENCODER CONFIGURATION

Test Label: Media Configuration NVT Video Encoder Configuration

ONVIF Core Specification Coverage: 10.2.2 Get media profiles, 10.5.1 Get video encoder configurations, 10.5.3 Get compatible video encoder configurations

Device Type: NVT

Command Under Test: None

WSDL Reference: media.wsdl

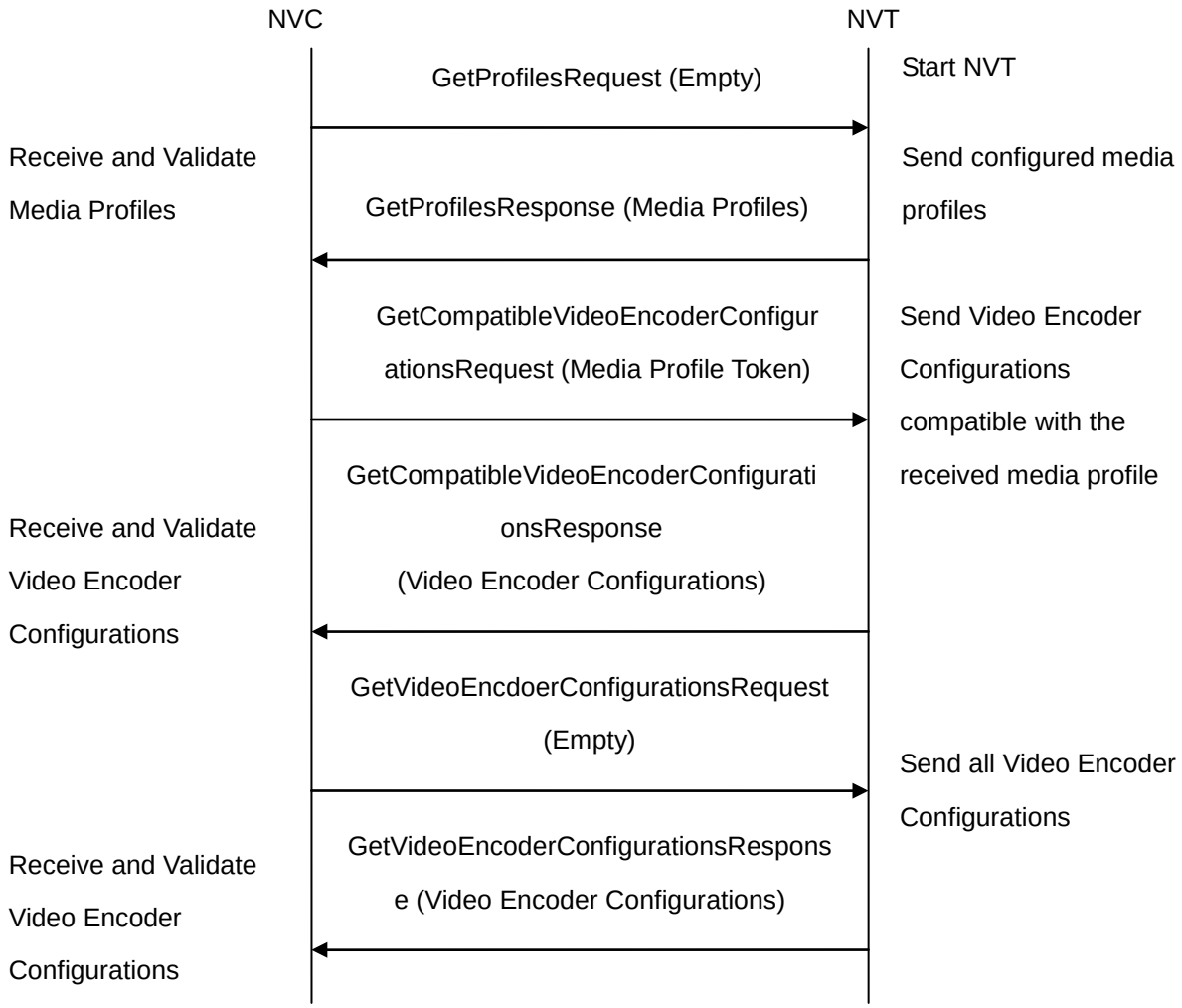
Requirement Level: MUST

Test Purpose: To verify NVT Video Encoder Configuration Operations

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes GetProfiles Request.
4. NVT sends the list of existing media profiles in GetProfiles Response message.
5. NVC invokes GetCompatibleVideoEncoderConfigurations request with one of the received media profile tokens as input argument.
6. NVT sends the list of video encoder configurations, compatible with the received media profile token.
7. NVC verifies the list of video encoder configurations sent by NVT.
8. NVC will invoke GetVideoEncoderConfigurations request to retrieve the list of video encoder configurations supported by the NVT.
9. NVT sends the list of all video encoder configurations supported by it.

10. NVC verifies the list of video encoder configurations sent by the NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send GetProfilesResponse message.

DUT did not send valid GetCompatibleVideoEncoderConfigurationsResponse message.

DUT did not send valid GetVideoEncoderConfigurationsResponse message.

7.2.3 NVT JPEG VIDEO ENCODER CONFIGURATION

Test Label: Media Configuration NVT JPEG Video Encoder Configuration

ONVIF Core Specification Coverage: 10.5.1 Get video encoder configurations, 10.5.2 Get video encoder configuration, 10.5.4 Get video encoder configuration options, 10.5.5 Modify a video encoder configuration.

Device Type: NVT

Command Under Test: None

WSDL Reference: media.wsdl

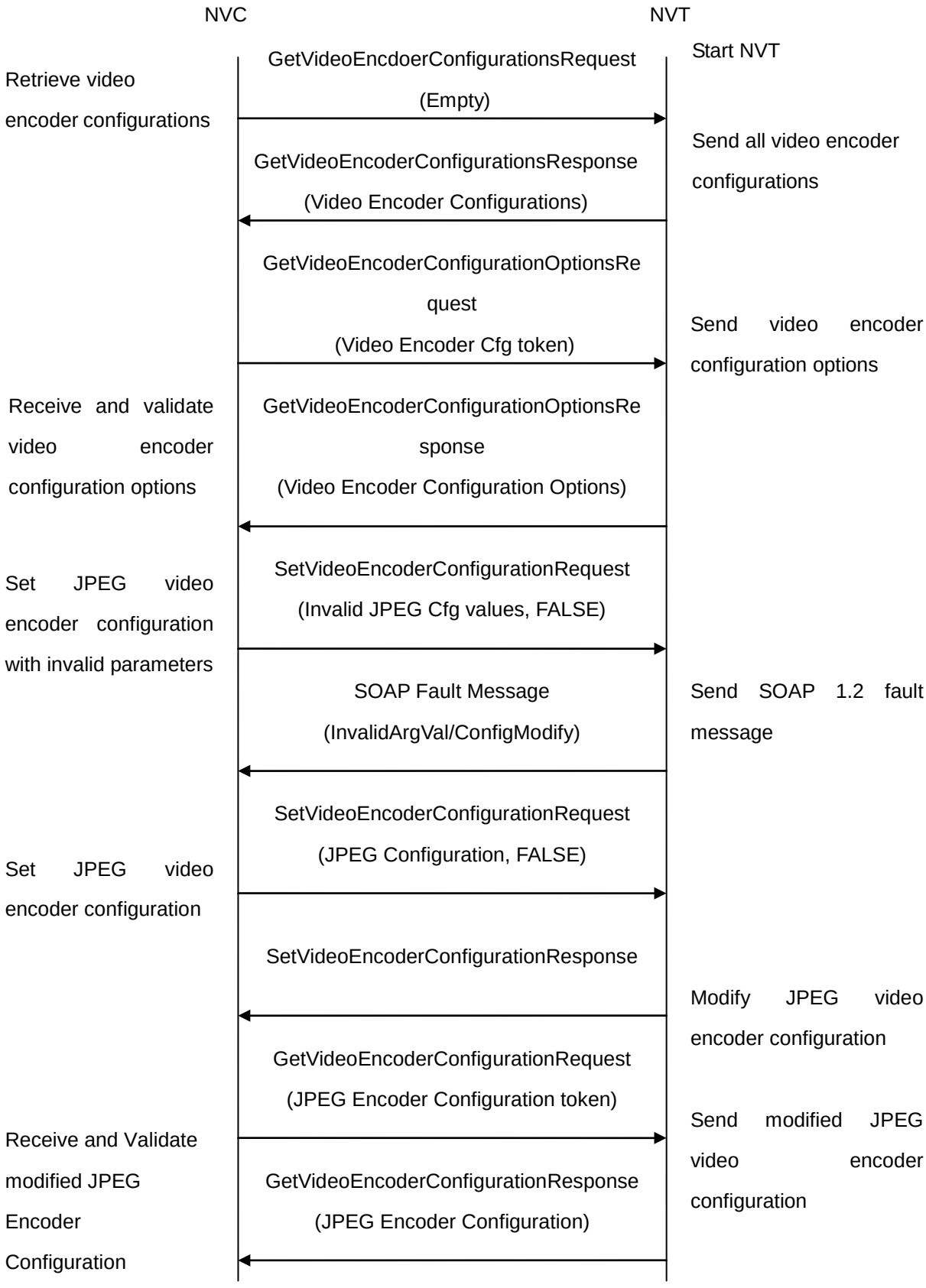
Requirement Level: MUST

Test Purpose: To verify NVT JPEG Video Encoder Configurations Setting

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Retrieve video encoder configurations
GetVideoEncdoerConfigurationsRequest
(Empty)
Send all video encoder configurations
GetVideoEncoderConfigurationsResponse
(Video Encoder Configurations)
GetVideoEncoderConfigurationOptionsRequest
(Video Encoder Cfg token)
Send video encoder configuration options
Receive and validate video encoder configuration options
GetVideoEncoderConfigurationOptionsResponse
(Video Encoder Configuration Options)
Set JPEG video encoder configuration with invalid parameters
SetVideoEncoderConfigurationRequest
(Invalid JPEG Cfg values, FALSE)
SOAP Fault Message
(InvalidArgVal/ConfigModify)
Send SOAP 1.2 fault message
Set JPEG video encoder configuration
SetVideoEncoderConfigurationRequest
(JPEG Configuration, FALSE)
SetVideoEncoderConfigurationResponse
Modify JPEG video encoder configuration
GetVideoEncoderConfigurationRequest
(JPEG Encoder Configuration token)
Send modified JPEG video encoder configuration
Receive and Validate modified JPEG Encoder Configuration
GetVideoEncoderConfigurationResponse
(JPEG Encoder Configuration)

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes GetVideoEncoderConfigurationsRequest to retrieve the list of video encoder configurations supported by NVT.
4. NVT sends video encoder configurations in the GetVideoEncoderConfigurationsResponse message.
5. NVC invokes GetVideoEncoderConfigurationOptions Request (Video Encoder Configuration token) to retrieve video encoder configuration options for the specified video encoder configuration.
6. NVT sends the range of configurable values for the received video encoder configuration in the GetVideoEncoderConfigurationOptionsResponse message.
7. Test steps -5 & 6 have to be repeated for all video encoder configurations until NVC finds a video encoder configuration with JPEG encoding support.
8. NVC invokes SetVideoEncoderConfiguration request with JPEG configuration values outside the range defined in the GetVideoEncoderConfigurationOptionsResponse and 'ForcePersistence' flag as 'FALSE'.
9. NVT send the SOAP 1.2 fault message (InvalidArgVal/ConfigModify)
10. NVC verifies the SOAP fault message sent by NVT.
11. NVC invokes SetVideoEncoderConfiguration request (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1 and force persistence = false**). These values will be taken from GetVideoEncoderConfigurationOptionsResponse message.
12. NVT modifies JPEG video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
13. NVC verifies the JPEG Video Encoder Configuration settings on NVT by invoking GetVideoEncoderConfiguration request.
14. NVT sends modified JPEG Video Encoder Configuration in the GetVideoEncoderConfigurationResponse message (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1**).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send GetVideoEncoderConfigurationsResponse message.

DUT did not send GetVideoEncoderConfigurationOptionsResponse message.

DUT doesn't support JPEG encoding.

DUT did not send the SOAP 1.2 fault message (InvalidArgVal/ConfigModify) for invalid SetVideoEncoderConfiguration request.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetVideoEncoderConfigurationResponse message.

The DUT did not modify JPEG Video Encoder Configuration.

7.2.4 NVT MPEG4 VIDEO ENCODER CONFIGURATION

Test Label: Media Configuration NVT MPEG4 Video Encoder Configuration

ONVIF Core Specification Coverage: 10.5.1 Get video encoder configurations, 10.5.2 Get video encoder configuration, 10.5.4 Get video encoder configuration options, 10.5.5 Modify a video encoder configuration.

Device Type: NVT

Command Under Test: None

WSDL Reference: media.wsdl

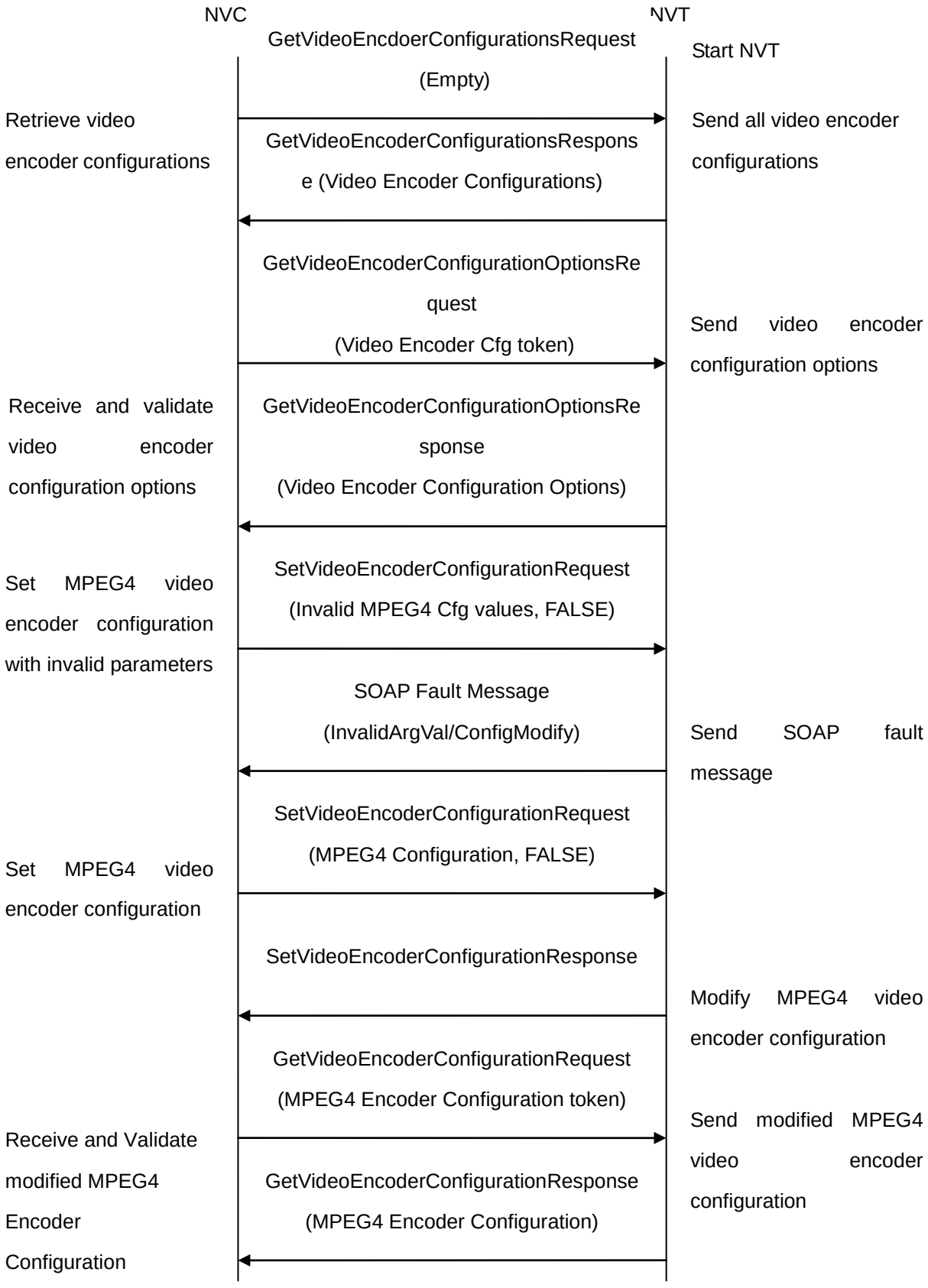
Requirement Level: MUST IF IMPLEMENTED (MPEG4-SP)

Test Purpose: To verify NVT MPEG4 Video Encoder Configurations Setting

Pre-Requisite: MPEG4 is implemented by NVT

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
GetVideoEncdoerConfigurationsRequest
(Empty)
Start NVT
Retrieve video encoder configurations
GetVideoEncoderConfigurationsResponse (Video Encoder Configurations)
Send all video encoder configurations
GetVideoEncoderConfigurationOptionsRequest
(Video Encoder Cfg token)
Send video encoder configuration options
Receive and validate video encoder configuration options
GetVideoEncoderConfigurationOptionsResponse
(Video Encoder Configuration Options)
Set MPEG4 video encoder configuration with invalid parameters
SetVideoEncoderConfigurationRequest
(Invalid MPEG4 Cfg values, FALSE)
SOAP Fault Message
(InvalidArgVal/ConfigModify)
Send SOAP fault message
Set MPEG4 video encoder configuration
SetVideoEncoderConfigurationRequest
(MPEG4 Configuration, FALSE)
SetVideoEncoderConfigurationResponse
Modify MPEG4 video encoder configuration
GetVideoEncoderConfigurationRequest
(MPEG4 Encoder Configuration token)
Send modified MPEG4 video encoder configuration
Receive and Validate modified MPEG4 Encoder Configuration
GetVideoEncoderConfigurationResponse
(MPEG4 Encoder Configuration)

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes GetVideoEncoderConfigurationsRequest to retrieve the list of video encoder configurations supported by NVT.
4. NVT sends video encoder configurations in the GetVideoEncoderConfigurationsResponse message.
5. NVC invokes GetVideoEncoderConfigurationOptions Request (Video Encoder Configuration token) to retrieve video encoder configuration options for the specified video encoder configuration.
6. NVT sends the range of configurable values for the received video encoder configuration in the GetVideoEncoderConfigurationOptionsResponse message.
7. Test steps -5 & 6 have to be repeated for all video encoder configurations until NVC finds a video encoder configuration with MPEG4 encoding support.
8. NVC invokes SetVideoEncoderConfiguration request with MPEG4 configuration values outside the range defined in the GetVideoEncoderConfigurationOptionsResponse and 'ForcePersistence' flag as 'FALSE'.
9. NVT send the SOAP 1.2 fault message (InvalidArgVal/ConfigModify)
10. NVC verifies the SOAP fault message sent by NVT.
11. NVC invokes SetVideoEncoderConfiguration request (**Encoding = "MPEG4", Resolution = ["Width", "Height"], Quality = q1, GovLength = g1, Mpeg4Profile = SP and force persistence = false**). These values will be taken from GetVideoEncoderConfigurationOptionsResponse message.
12. NVT modifies MPEG4 video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
13. NVC verifies the MPEG4 Video Encoder Configuration settings on NVT by GetVideoEncoderConfigurationRequest message.
14. NVT sends modified MPEG4 Video Encoder Configuration in the GetVideoEncoderConfigurationResponse message (**Encoding = "MPEG4", Resolution = ["Width", "Height"], Quality = q1, GovLength = g1, Mpeg4Profile = SP**).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send GetVideoEncoderConfigurationsResponse message.

DUT did not send GetVideoEncoderConfigurationOptionsResponse message.

DUT doesn't support MPEG4 encoding.

DUT did not send the SOAP 1.2 fault message (InvalidArgVal/ConfigModify) for invalid SetVideoEncoderConfiguration request.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetVideoEncoderConfigurationResponse message.

The DUT did not modify MPEG4 Video Encoder Configuration.

7.2.5 NVT H.264 VIDEO ENCODER CONFIGURATION

Test Label: Media Configuration NVT H.264 Video Encoder Configuration

ONVIF Core Specification Coverage: 10.5.1 Get video encoder configurations, 10.5.2 Get video encoder configuration, 10.5.4 Get video encoder configuration options, 10.5.5 Modify a video encoder configuration.

Device Type: NVT

Command Under Test: None

WSDL Reference: media.wsdl

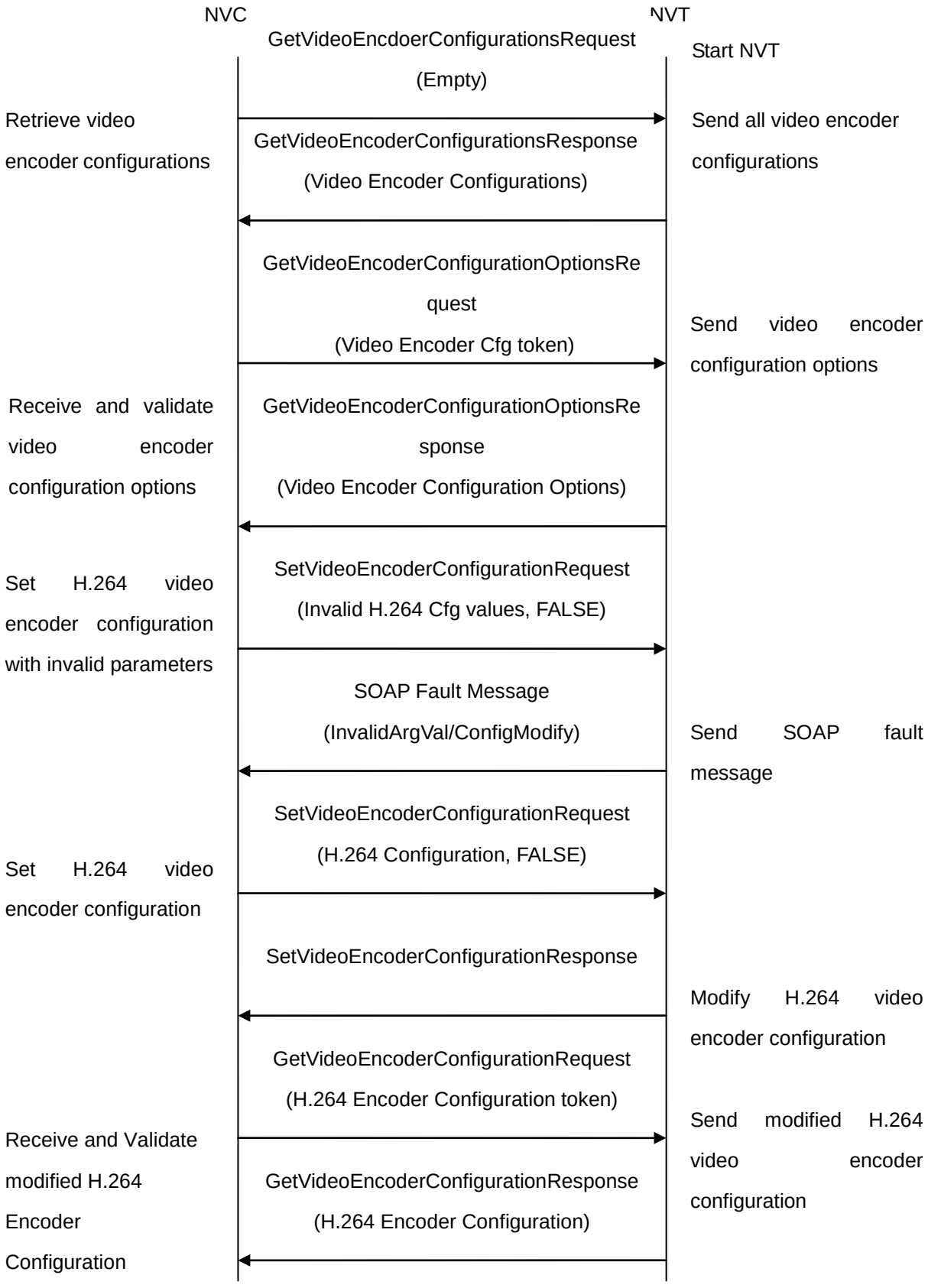
Requirement Level: MUST IF IMPLEMENTED (H.264-Baseline)

Test Purpose: To verify NVT H.264 Video Encoder Configurations Setting

Pre-Requisite: H.264 is implemented by NVT

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
GetVideoEncdoerConfigurationsRequest
(Empty)
Start NVT
Retrieve video encoder configurations
GetVideoEncoderConfigurationsResponse
(Video Encoder Configurations)
Send all video encoder configurations
GetVideoEncoderConfigurationOptionsRequest
(Video Encoder Cfg token)
Send video encoder configuration options
Receive and validate video encoder configuration options
GetVideoEncoderConfigurationOptionsResponse
(Video Encoder Configuration Options)
Set H.264 video encoder configuration with invalid parameters
SetVideoEncoderConfigurationRequest
(Invalid H.264 Cfg values, FALSE)
SOAP Fault Message
(InvalidArgVal/ConfigModify)
Send SOAP fault message
Set H.264 video encoder configuration
SetVideoEncoderConfigurationRequest
(H.264 Configuration, FALSE)
SetVideoEncoderConfigurationResponse
Modify H.264 video encoder configuration
GetVideoEncoderConfigurationRequest
(H.264 Encoder Configuration token)
Receive and Validate modified H.264 Encoder Configuration
Send modified H.264 video encoder configuration
GetVideoEncoderConfigurationResponse
(H.264 Encoder Configuration)

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes GetVideoEncoderConfigurationsRequest to retrieve the list of video encoder configurations supported by NVT.
4. NVT sends video encoder configurations in the GetVideoEncoderConfigurationsResponse message.
5. NVC invokes GetVideoEncoderConfigurationOptions Request (Video Encoder Configuration token) to retrieve video encoder configuration options for the specified video encoder configuration.
6. NVT sends the range of configurable values for the received video encoder configuration in the GetVideoEncoderConfigurationOptionsResponse message.
7. Test steps -5 & 6 have to be repeated for all video encoder configurations until NVC finds a video encoder configuration with H.264 encoding support
8. NVC invokes SetVideoEncoderConfiguration request with H.264 configuration values outside the range defined in the GetVideoEncoderConfigurationOptionsResponse and 'ForcePersistence' flag as 'FALSE'.
9. NVT send the SOAP 1.2 fault message (InvalidArgVal/ConfigModify)
10. NVC verifies the SOAP fault message sent by NVT.
11. NVC will invoke SetVideoEncoderConfiguration request (**Encoding = "H.264", Resolution = ["Width", "Height"], Quality = q1, GovLength = g1, H264Profile = "Baseline" and force persistence = false**). These values will be taken from GetVideoEncoderConfigurationOptionsResponse message.
12. NVT modifies H.264 video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
13. NVC will verify the H.264 Video Encoder Configuration settings on NVT by GetVideoEncoderConfigurationRequest message.
14. NVT sends modified H.264 Video Encoder Configurations in the GetVideoEncoderConfigurationResponse message (**Encoding = "H.264", Resolution = ["Width", "Height"], Quality = q1, GovLength = g1, H264Profile = "Baseline"**).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send GetVideoEncoderConfigurationsResponse.

DUT did not send GetVideoEncoderConfigurationOptionsResponse message.

DUT doesn't support H.264 encoding.

DUT did not send the SOAP 1.2 fault message (InvalidArgVal/ConfigModify) for invalid SetVideoEncoderConfiguration request.

DUT did not send GetVideoEncoderConfigurationResponse message.

DUT did not send SetVideoEncoderConfigurationResponse message.

The DUT did not modify H.264 Video Encoder Configuration.

7.2.6 NVT GUARANTEED NUMBER OF VIDEO ENCODER INSTANCES

Test Label: Media Configuration NVT Video Encoder Instances

ONVIF Core Specification Coverage: 10.4.1 Get video source configurations, 10.5.6 Get guaranteed number of video encoder instances

Device Type: NVT

Command Under Test: GetGuaranteedNumberOfVideoEncoderInstances

WSDL Reference: media.wsdl

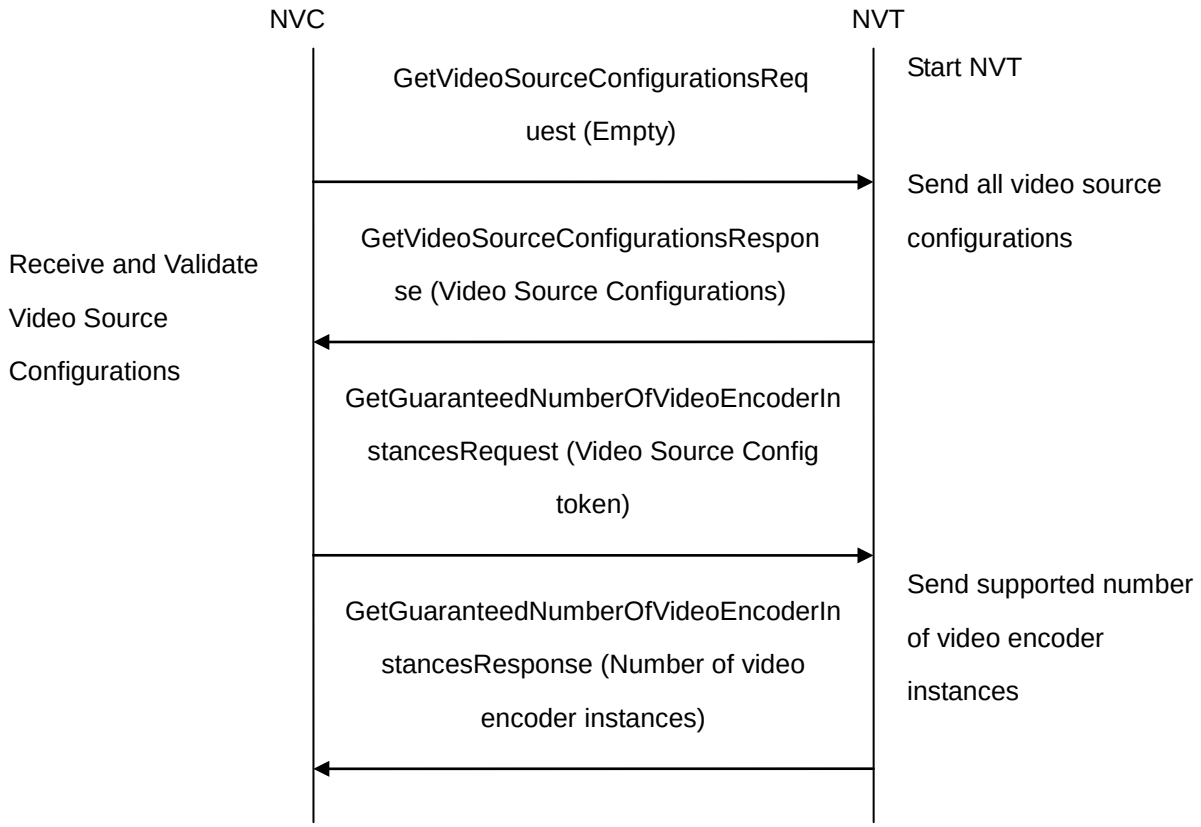
Requirement Level: MUST

Test Purpose: To retrieve minimum number of video encoder instances supported by NVT per video source configuration.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes GetVideoSourceConfigurations request.
4. NVT sends the list of video source configurations supported by it.
5. NVC invokes GetGuaranteedNumberOfVideoEncoderInstancesRequest message for a selected video source configuration.
6. NVT sends the minimum guaranteed number of video encoder instances.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send GetVideoSourceConfigurationsResponse message.

DUT did not send GetGuaranteedNumberOfVideoEncoderInstancesResponse message.

DUT did not send ‘TotalNumber’ value.

7.3 Audio Configuration

7.3.1 NVT AUDIO SOURCE CONFIGURATION

Test Label: Media Configuration NVT Audio Source Configuration

ONVIF Core Specification Coverage: 10.2.1 Create media profile, 10.2.6 Add audio source configuration to a profile, 10.2.13 Remove audio source configuration from a profile, 10.2.18 Delete media profile, 10.6.1 Get audio sources, 10.7.1 Get audio source configurations, 10.7.2 Get audio source configuration, 10.7.3 Get compatible audio source configurations, 10.7.4 Get audio source configuration options, 10.7.5 Modify an audio source configuration.

Device Type: NVT

Command Under Test: None

WSDL Reference: media.wsdl

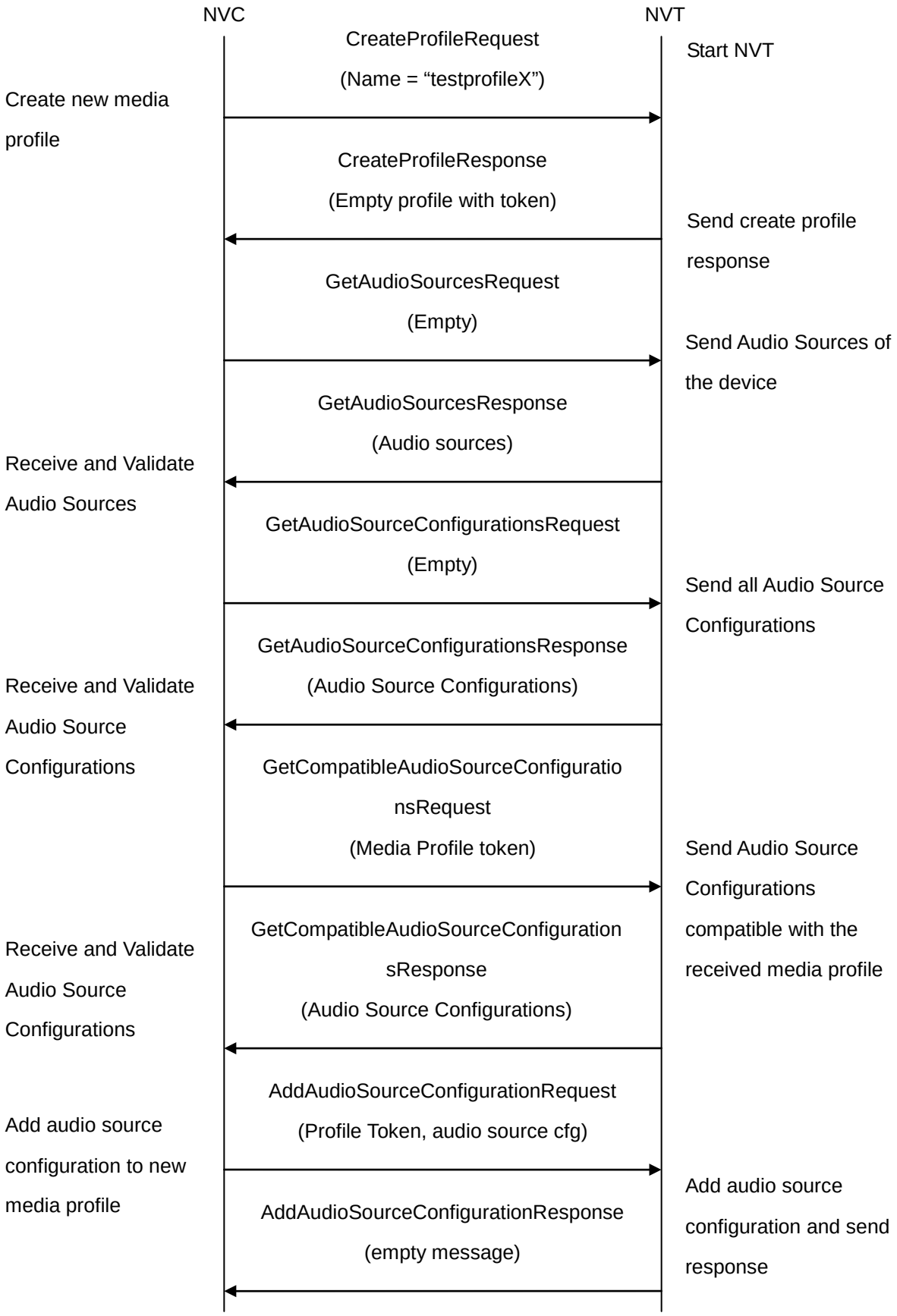
Requirement Level: MUST IF SUPPORTED (Audio)

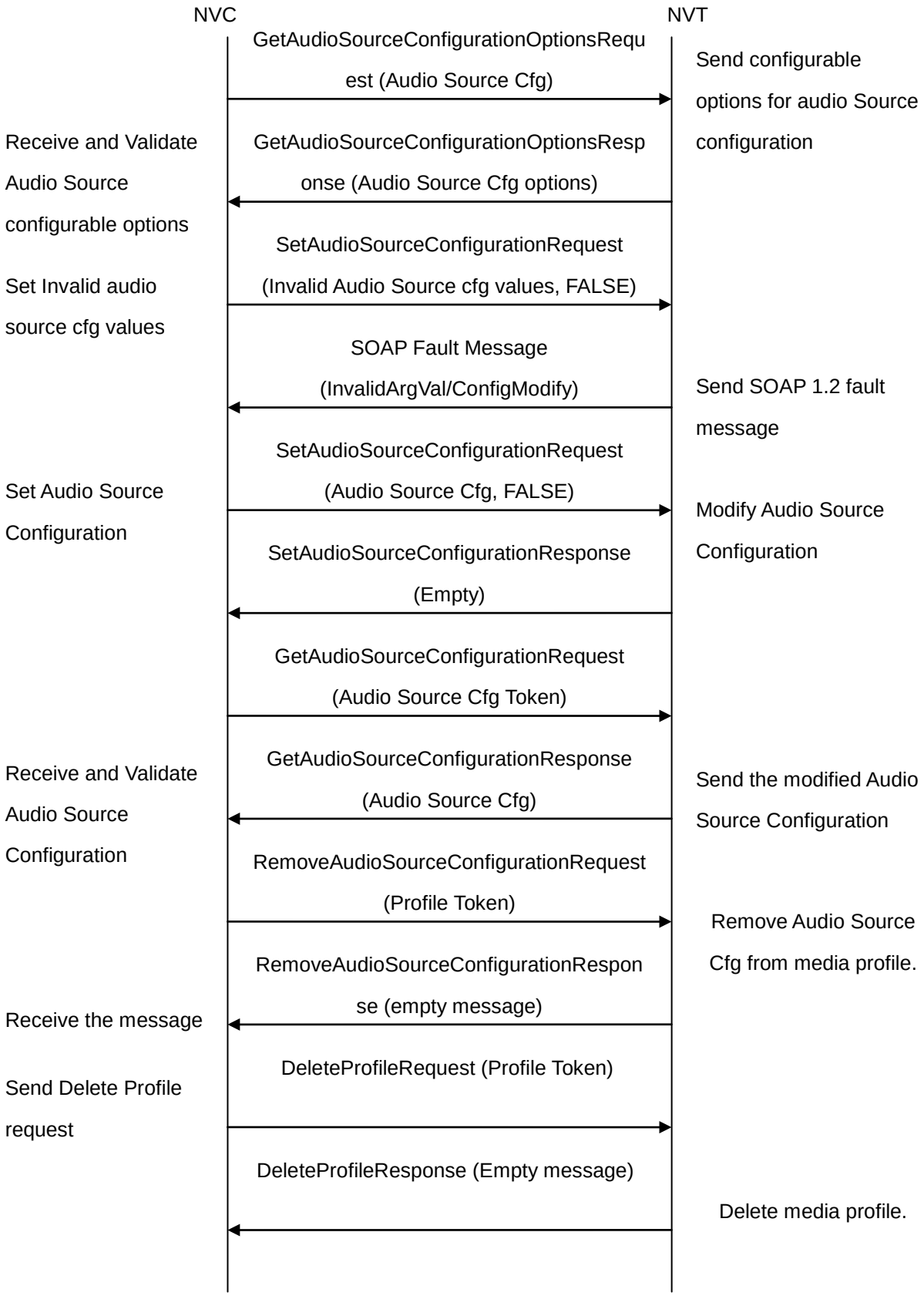
Test Purpose: To verify NVT Audio Source Configuration Operations

Pre-Requisite: Audio is supported by NVT

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
CreateProfileRequest
(Name = “testprofileX”)
Start NVT
Create new media profile
CreateProfileResponse
(Empty profile with token)
Send create profile response
GetAudioSourcesRequest
(Empty)
Send Audio Sources of the device
GetAudioSourcesResponse
(Audio sources)
Receive and Validate Audio Sources
GetAudioSourceConfigurationsRequest
(Empty)
Send all Audio Source Configurations
GetAudioSourceConfigurationsResponse
(Audio Source Configurations)
Receive and Validate Audio Source Configurations
GetCompatibleAudioSourceConfiguratio nsRequest
(Media Profile token)
Send Audio Source Configurations compatible with the received media profile
GetCompatibleAudioSourceConfiguration sResponse
(Audio Source Configurations)
Receive and Validate Audio Source Configurations
AddAudioSourceConfigurationRequest
(Profile Token, audio source cfg)
Add audio source configuration to new media profile
AddAudioSourceConfigurationResponse
(empty message)
Add audio source configuration and send response


NVC
NVT
GetAudioSourceConfigurationOptionsRequest (Audio Source Cfg)
Send configurable options for audio Source configuration
Receive and Validate Audio Source configurable options
GetAudioSourceConfigurationOptionsResponse (Audio Source Cfg options)
SetAudioSourceConfigurationRequest (Invalid Audio Source cfg values, FALSE)
Set Invalid audio source cfg values
SOAP Fault Message (InvalidArgVal/ConfigModify)
Send SOAP 1.2 fault message
SetAudioSourceConfigurationRequest (Audio Source Cfg, FALSE)
Set Audio Source Configuration
Modify Audio Source Configuration
SetAudioSourceConfigurationResponse (Empty)
GetAudioSourceConfigurationRequest (Audio Source Cfg Token)
GetAudioSourceConfigurationResponse (Audio Source Cfg)
Receive and Validate Audio Source Configuration
Send the modified Audio Source Configuration
RemoveAudioSourceConfigurationRequest (Profile Token)
Remove Audio Source Cfg from media profile.
RemoveAudioSourceConfigurationResponse (empty message)
Receive the message
DeleteProfileRequest (Profile Token)
Send Delete Profile request
DeleteProfileResponse (Empty message)
Delete media profile.

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes CreateProfile request with **ProfileToken** = **'testprofileX'**.
4. NVT creates new media profile and sends the response.
5. NVC invokes GetAudioSources request to retrieve the existing audio sources of NVT.
6. NVC verifies the list of audio sources sent by NVT.
7. NVC invokes GetAudioSourceConfigurations request to retrieve the list of audio source configurations supported by the NVT.
8. NVC verifies the list of audio source configurations sent by NVT.
9. NVC invokes GetCompatibleAudioSourceConfigurations request with **'testprofileX'** as **ProfileToken**.
10. NVT sends the list of audio source configurations compatible with the received media profile token.
11. NVC invokes AddAudioSourceConfiguration request message with **ProfileToken** as **'testprofileX'** and **ConfigurationToken** as one of the tokens in GetCompatibleAudioSourceConfigurations response.
12. NVT adds the audio source configuration to the media profile and sends the response.
13. NVC invokes GetAudioSourceConfigurationOptions request with **ConfigurationToken** as the same token sent in the AddAudioSourceConfiguration request.
14. NVT sends the configurable options supported for the received audio source configuration.
15. NVC invokes SetAudioSourceConfiguration request with audio source configuration values outside the range defined in GetAudioSourceConfigurationOptions response and **'ForcePersistence'** flag as **'FALSE'**.
16. NVT send the SOAP 1.2 fault message **(InvalidArgVal/ConfigModify)**.
17. NVC verifies the SOAP fault message sent by NVT.
18. NVC invokes SetAudioSourceConfiguration request with audio source configuration values as defined in GetAudioSourceConfigurationOptions response and **'ForcePersistence'** flag as **'FALSE'**.
19. NVT modifies the audio source configuration and sends the SetAudioSourceConfigurationResponse message indicating success.
20. NVC verifies the modified audio source configuration by invoking the GetAudioSourceConfiguration request.
21. NVT sends the modified audio source configuration in GetAudioSourceConfiguration response.
22. NVC invokes RemoveAudioSourceConfiguration request with **ProfileToken** as **'testprofileX'**.
23. NVT removes the audio source configuration token from media profile and sends the response.

24. NVC invokes DeleteProfile request with **ProfileToken** as **'testprofileX'**.
25. NVT deletes the media profile and sends the response.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send CreateProfileResponse message.

DUT did not send valid GetAudioSourcesResponse message.

DUT did not send valid GetAudioSourceConfigurationsResponse message.

DUT did not send GetCompatibleAudioSourceConfigurationsResponse message.

DUT did not send AddAudioSourceConfigurationResponse message.

DUT did not send GetAudioSourceConfigurationOptionsResponse message.

DUT did not send the SOAP 1.2 fault message (InvalidArgVal/ConfigModify) for invalid SetAudioSourceConfiguration request.

DUT did not send SetAudioSourceConfigurationResponse message.

DUT did not send GetAudioSourceConfigurationResponse message.

DUT did not modify audio source configuration correctly.

DUT did not send RemoveAudioSourceConfigurationResponse message.

DUT did not send DeleteProfileResponse message.

7.3.2 NVT AUDIO ENCODER CONFIGURATION

Test Label: Media Configuration NVT Audio Encoder Configuration

ONVIF Core Specification Coverage: 10.2.1 Create media profile, 10.2.6 Add audio source configuration to a profile, 10.2.7 Add audio encoder configuration to a profile, 10.2.13 Remove audio source configuration from a profile, 10.2.14 Remove audio encoder configuration from a profile, 10.2.18 Delete media profile, 10.7.1 Get audio source configurations, 10.8.1 Get audio encoder configurations, 10.8.3 Get compatible audio encoder configurations.

Device Type: NVT

Command Under Test: None

WSDL Reference: media.wsdl

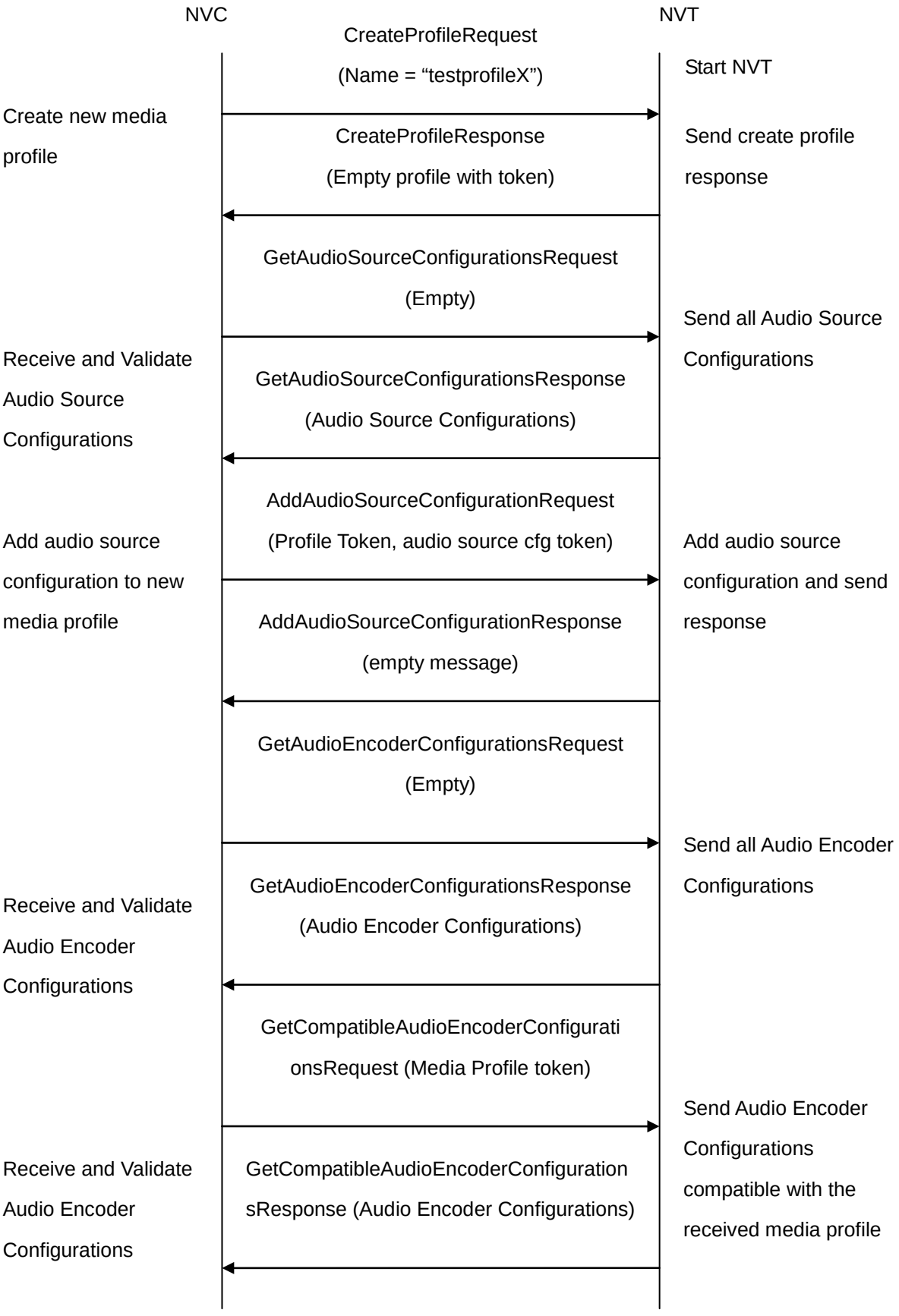
Requirement Level: MUST IF SUPPORTED (Audio)

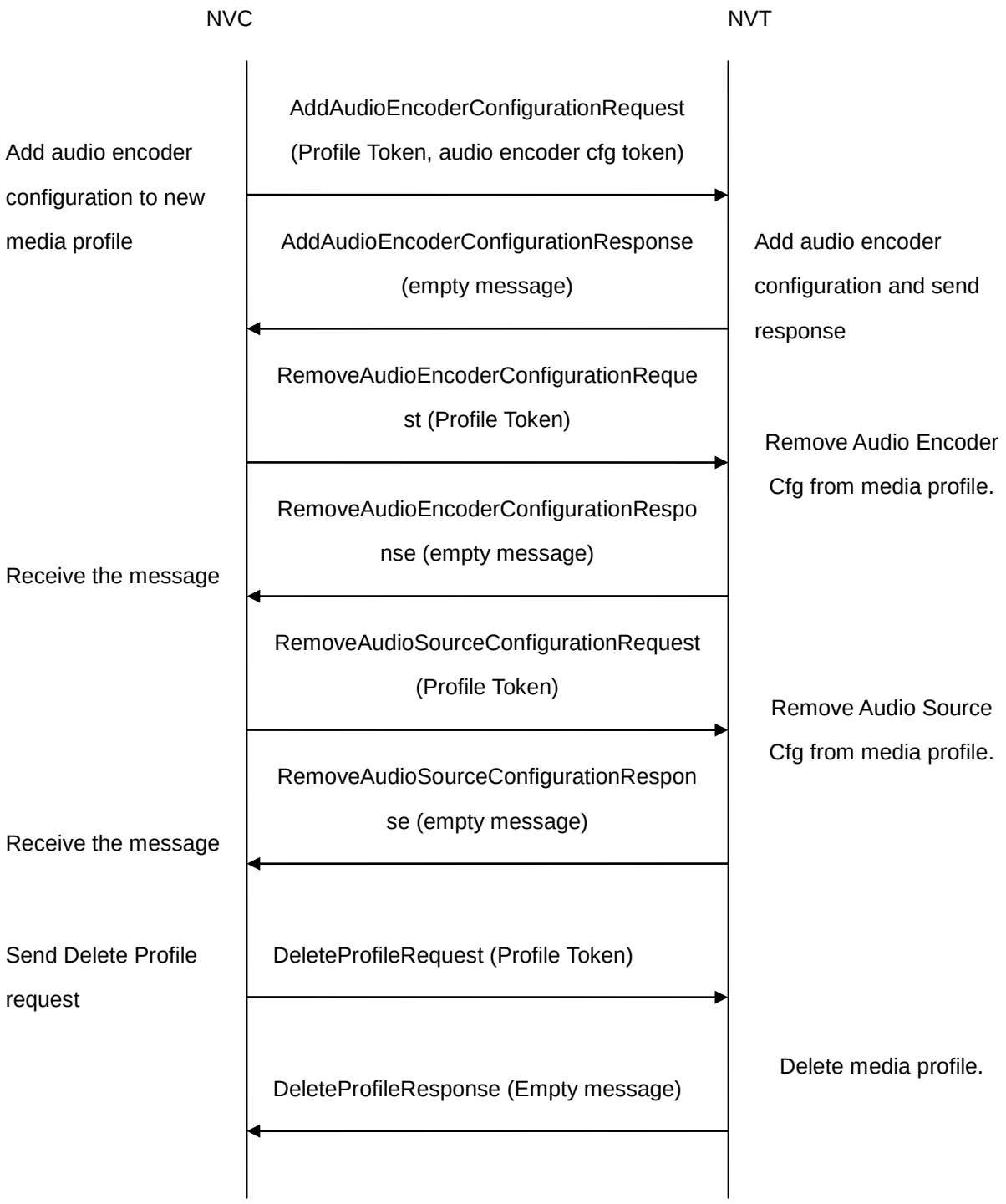
Test Purpose: To verify NVT Audio Encoder Configuration Operations

Pre-Requisite: Audio is supported by NVT

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
CreateProfileRequest
(Name = “testprofileX”)
Start NVT
Create new media profile
CreateProfileResponse
(Empty profile with token)
Send create profile response
GetAudioSourceConfigurationsRequest
(Empty)
Send all Audio Source Configurations
Receive and Validate Audio Source Configurations
GetAudioSourceConfigurationsResponse
(Audio Source Configurations)
AddAudioSourceConfigurationRequest
(Profile Token, audio source cfg token)
Add audio source configuration to new media profile
Add audio source configuration and send response
AddAudioSourceConfigurationResponse
(empty message)
GetAudioEncoderConfigurationsRequest
(Empty)
Send all Audio Encoder Configurations
GetAudioEncoderConfigurationsResponse
(Audio Encoder Configurations)
Receive and Validate Audio Encoder Configurations
GetCompatibleAudioEncoderConfigurationsRequest (Media Profile token)
Send Audio Encoder Configurations compatible with the received media profile
Receive and Validate Audio Encoder Configurations
GetCompatibleAudioEncoderConfigurationsResponse (Audio Encoder Configurations)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes CreateProfile request with **ProfileToken** = **'testprofileX'**.

4. NVT creates new media profile and sends the response.
5. NVC will invoke GetAudioSourceConfigurations request to retrieve the list of audio source configurations supported by NVT.
6. NVC verifies the list of audio source configurations sent by NVT.
7. NVC invokes AddAudioSourceConfiguration request message with **ProfileToken** as **'testprofileX'** and **ConfigurationToken** as one of the tokens received in the GetAudioSourceConfigurations response.
8. NVT adds the audio source configuration to the profile and sends the response.
9. NVC will invoke GetAudioEncoderConfigurations request to retrieve the list of audio encoder configurations supported by NVT.
10. NVC verifies the list of audio encoder configurations sent by NVT.
11. NVC invokes GetCompatibleAudioEncoderConfigurations request with **'testprofileX'** as **ProfileToken**.
12. NVT sends the list of audio encoder configurations compatible with the received media profile token.
13. NVC invokes AddAudioEncoderConfiguration request message with **ProfileToken** as **'testprofileX'** and **ConfigurationToken** as one of the tokens received in the GetCompatibleAudioEncoderConfigurations response.
14. NVT adds the audio encoder configuration to the profile and sends the response.
15. NVC invokes RemoveAudioEncoderConfiguration request with **ProfileToken** as **'testprofileX'**.
16. NVT removes the audio encoder configuration token from media profile and sends the response.
17. NVC invokes RemoveAudioSourceConfiguration request with **ProfileToken** as **'testprofileX'**.
18. NVT removes the audio source configuration token from media profile and sends the response.
19. NVC invokes DeleteProfile request with **ProfileToken** as **'testprofileX'**.
20. NVT deletes the media profile and sends the response.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send CreateProfileResponse message.

DUT did not send valid GetAudioSourceConfigurationsResponse message.

DUT did not send AddAudioSourceConfigurationResponse message.

DUT did not send valid GetAudioEncoderConfigurationsResponse message.

DUT did not send GetCompatibleAudioEncoderConfigurationsResponse message.

DUT did not send AddAudioEncoderConfigurationResponse message.

DUT did not send RemoveAudioEncoderConfigurationResponse message.

DUT did not send RemoveAudioSourceConfigurationResponse message.

DUT did not send DeleteProfileResponse message.

7.3.3 NVT G.711 AUDIO ENCODER CONFIGURATION

Test Label: Media Configuration NVT G.711 Audio Encoder Configuration

ONVIF Core Specification Coverage: 10.8.1 Get audio encoder configurations, 10.8.2 Get audio encoder configuration, 10.8.4 Get audio encoder configuration options, 10.8.5 Modify audio encoder configurations

Device Type: NVT

Command Under Test: None

WSDL Reference: media.wsdl

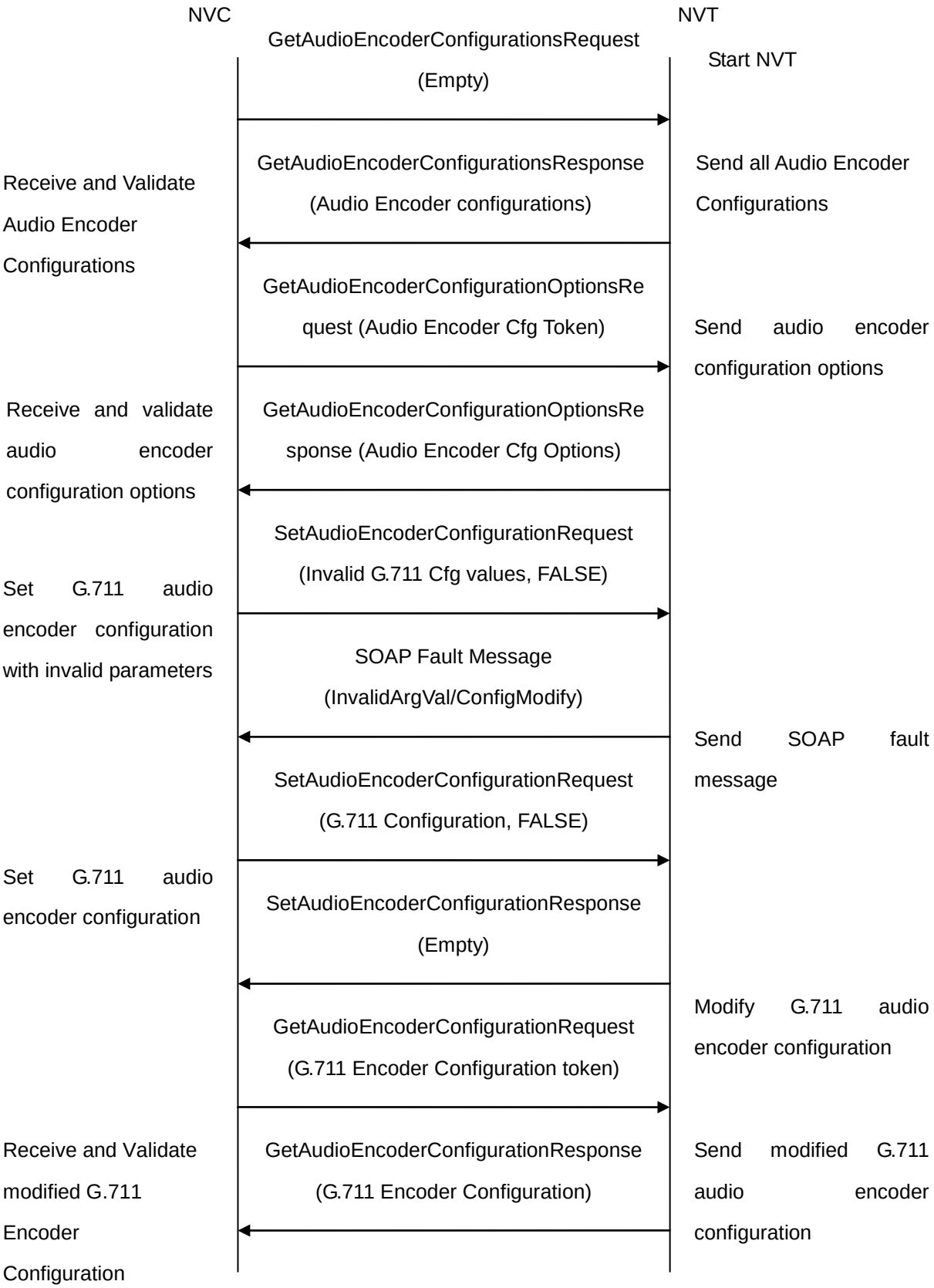
Requirement Level: MUST IF SUPPORTED (Audio)

Test Purpose: To verify NVT G.711 Audio Encoder Configuration Setting

Pre-Requisite: Audio is supported by NVT

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
GetAudioEncoderConfigurationsRequest
(Empty)
Start NVT
Receive and Validate Audio Encoder Configurations
GetAudioEncoderConfigurationsResponse
(Audio Encoder configurations)
Send all Audio Encoder Configurations
GetAudioEncoderConfigurationOptionsRequest (Audio Encoder Cfg Token)
Send audio encoder configuration options
Receive and validate audio encoder configuration options
GetAudioEncoderConfigurationOptionsResponse (Audio Encoder Cfg Options)
Set G.711 audio encoder configuration with invalid parameters
SetAudioEncoderConfigurationRequest
(Invalid G.711 Cfg values, FALSE)
SOAP Fault Message
(InvalidArgVal/ConfigModify)
Send SOAP fault message
SetAudioEncoderConfigurationRequest
(G.711 Configuration, FALSE)
Set G.711 audio encoder configuration
SetAudioEncoderConfigurationResponse
(Empty)
Modify G.711 audio encoder configuration
GetAudioEncoderConfigurationRequest
(G.711 Encoder Configuration token)
Receive and Validate modified G.711 Encoder Configuration
GetAudioEncoderConfigurationResponse
(G.711 Encoder Configuration)
Send modified G.711 audio encoder configuration

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes GetAudioEncoderConfigurations request.
4. NVT sends the list of supported audio encoder configurations in GetAudioEncoderConfigurationsResponse message.
5. NVC invokes GetAudioEncoderConfigurationOptions Request (Audio Encoder Configuration token) to retrieve audio encoder configuration options for the specified audio encoder configuration.
6. NVT sends the range of configurable values for the received audio encoder configuration in the GetAudioEncoderConfigurationOptionsResponse message.
7. Test steps -5 & 6 have to be repeated for all audio encoder configurations until NVC finds a audio encoder configuration with G.711 encoding support
8. NVC invokes SetAudioEncoderConfiguration request with G.711 configuration values outside the range defined in the GetAudioEncoderConfigurationOptionsResponse and **'ForcePersistence'** flag as **'FALSE'**.
9. NVT send the SOAP 1.2 fault message **(InvalidArgVal/ConfigModify)**.
10. NVC verifies the SOAP fault message sent by NVT.
11. NVC invokes SetAudioEncoderConfiguration request (**Encoding = "G711", Bit Rate = r1, Sample Rate = r2 and force persistence = false**). These values will be taken from the GetAudioEncoderConfigurationOptionsResponse message.
12. NVT modifies G.711 audio encoder configuration and responds with SetAudioEncoderConfigurationResponse message indicating success.
13. NVC verifies the G.711 audio Encoder Configuration settings on NVT by invoking GetAudioEncoderConfiguration request.
14. NVT sends modified G.711 audio Encoder Configuration in the GetAudioEncoderConfigurationResponse message (**Encoding = "G711", Bit Rate = r1, Sample Rate = r2**).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send GetAudioEncoderConfigurationsResponse message.

DUT did not send GetAudioEncoderConfigurationOptionsResponse message.

DUT doesn't support G.711 audio encoding.

DUT did not send the SOAP 1.2 fault message (InvalidArgVal/ConfigModify) for invalid SetAudioEncoderConfiguration request.

DUT did not send SetAudioEncoderConfigurationResponse message.

DUT did not send GetAudioEncoderConfigurationResponse message.

The DUT did not modify G.711 Audio Encoder Configuration.

7.3.4 NVT G.726 AUDIO ENCODER CONFIGURATION

Test Label: Media Configuration NVT G.726 Audio Encoder Configuration

ONVIF Core Specification Coverage: 10.8.1 Get audio encoder configurations, 10.8.2 Get audio encoder configuration, 10.8.4 Get audio encoder configuration options, 10.8.5 Modify audio encoder configurations

Device Type: NVT

Command Under Test: None

WSDL Reference: media.wsdl

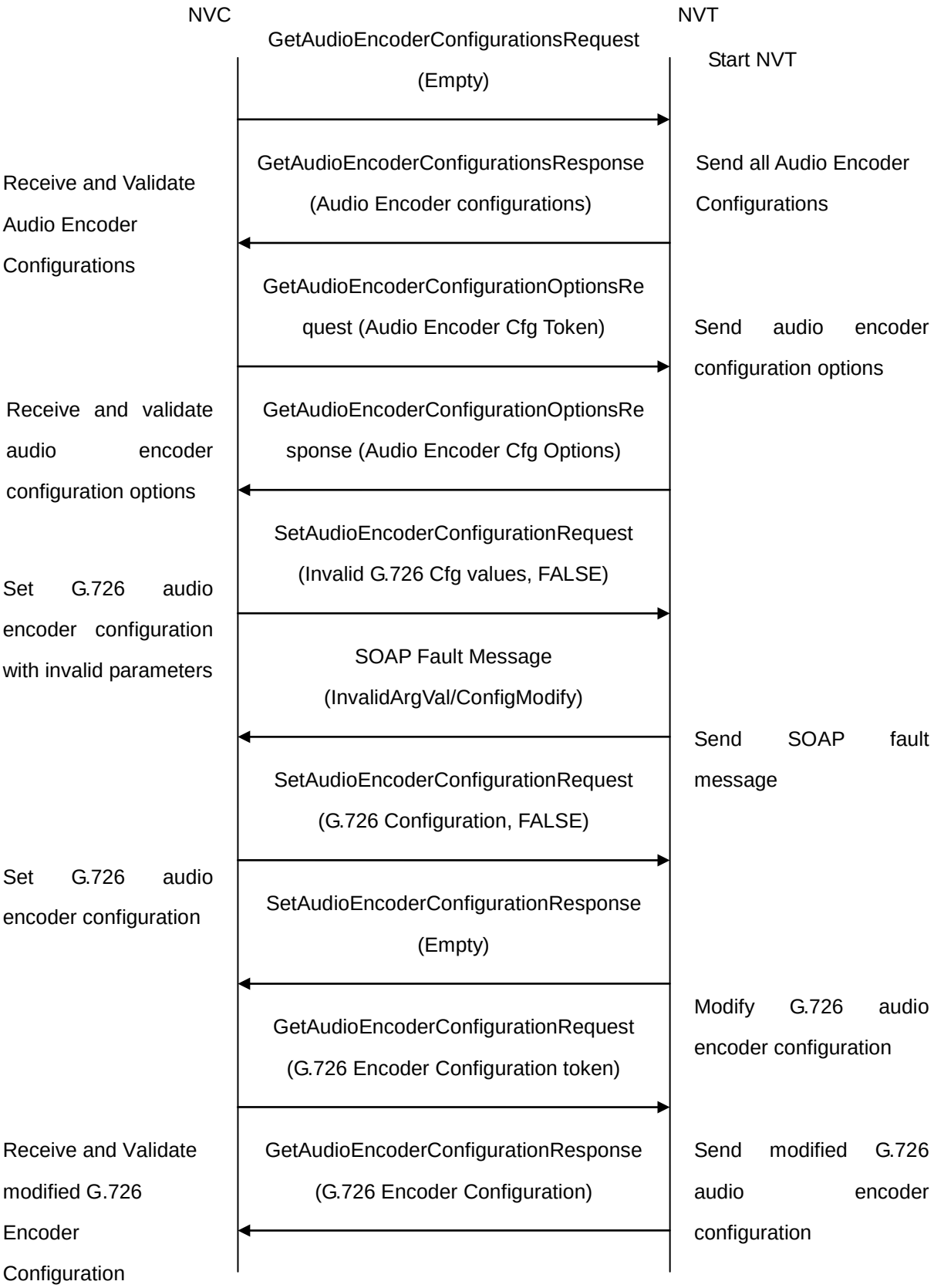
Requirement Level: MUST IF SUPPORTED (Audio) & IMPLEMENTED (G.726)

Test Purpose: To verify NVT G.726 Audio Encoder Configuration Setting

Pre-Requisite: Audio is supported by NVT and G.726 is implemented by NVT.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
GetAudioEncoderConfigurationsRequest
(Empty)
Start NVT
Receive and Validate Audio Encoder Configurations
GetAudioEncoderConfigurationsResponse
(Audio Encoder configurations)
Send all Audio Encoder Configurations
GetAudioEncoderConfigurationOptionsRequest (Audio Encoder Cfg Token)
Send audio encoder configuration options
Receive and validate audio encoder configuration options
GetAudioEncoderConfigurationOptionsResponse (Audio Encoder Cfg Options)
Set G.726 audio encoder configuration with invalid parameters
SetAudioEncoderConfigurationRequest
(Invalid G.726 Cfg values, FALSE)
SOAP Fault Message
(InvalidArgVal/ConfigModify)
Send SOAP fault message
SetAudioEncoderConfigurationRequest
(G.726 Configuration, FALSE)
Set G.726 audio encoder configuration
SetAudioEncoderConfigurationResponse
(Empty)
Modify G.726 audio encoder configuration
GetAudioEncoderConfigurationRequest
(G.726 Encoder Configuration token)
Receive and Validate modified G.726 Encoder Configuration
GetAudioEncoderConfigurationResponse
(G.726 Encoder Configuration)
Send modified G.726 audio encoder configuration

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes GetAudioEncoderConfigurations request.
4. NVT sends the list of supported audio encoder configurations in GetAudioEncoderConfigurationsResponse message.
5. NVC invokes GetAudioEncoderConfigurationOptions Request (Audio Encoder Configuration token) to retrieve audio encoder configuration options for the specified audio encoder configuration.
6. NVT sends the range of configurable values for the received audio encoder configuration in the GetAudioEncoderConfigurationOptionsResponse message.
7. Test steps -5 & 6 have to be repeated for all audio encoder configurations until NVC finds a audio encoder configuration with G.726 encoding support
8. NVC invokes SetAudioEncoderConfiguration request with G.726 configuration values outside the range defined in the GetAudioEncoderConfigurationOptionsResponse and **'ForcePersistence'** flag as **'FALSE'**.
9. NVT send the SOAP 1.2 fault message **(InvalidArgVal/ConfigModify)**.
10. NVC verifies the SOAP fault message sent by NVT.
11. NVC invokes SetAudioEncoderConfiguration request (**Encoding = "G726", Bit Rate = r1, Sample Rate = r2 and force persistence = false**). These values will be taken from the GetAudioEncoderConfigurationOptionsResponse message.
12. NVT modifies G.726 audio encoder configuration and responds with SetAudioEncoderConfigurationResponse message indicating success.
13. NVC verifies the G.726 audio Encoder Configuration settings on NVT by invoking GetAudioEncoderConfiguration request.
14. NVT sends the modified G.726 audio Encoder Configuration in the GetAudioEncoderConfigurationResponse message (**Encoding = "G726", Bit Rate = r1, Sample Rate = r2**).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send GetAudioEncoderConfigurationsResponse message.

DUT did not send GetAudioEncoderConfigurationOptionsResponse message.

DUT doesn't support G.726 audio encoding.

DUT did not send the SOAP 1.2 fault message (InvalidArgVal/ConfigModify) for invalid SetAudioEncoderConfiguration request.

DUT did not send SetAudioEncoderConfigurationResponse message.

DUT did not send GetAudioEncoderConfigurationResponse message.

The DUT did not modify G.726 Audio Encoder Configuration.

7.3.5 NVT AAC AUDIO ENCODER CONFIGURATION

Test Label: Media Configuration NVT AAC Audio Encoder Configuration

ONVIF Core Specification Coverage: 10.8.1 Get audio encoder configurations, 10.8.2 Get audio encoder configuration, 10.8.4 Get audio encoder configuration options, 10.8.5 Modify audio encoder configurations

Device Type: NVT

Command Under Test: None

WSDL Reference: media.wsdl

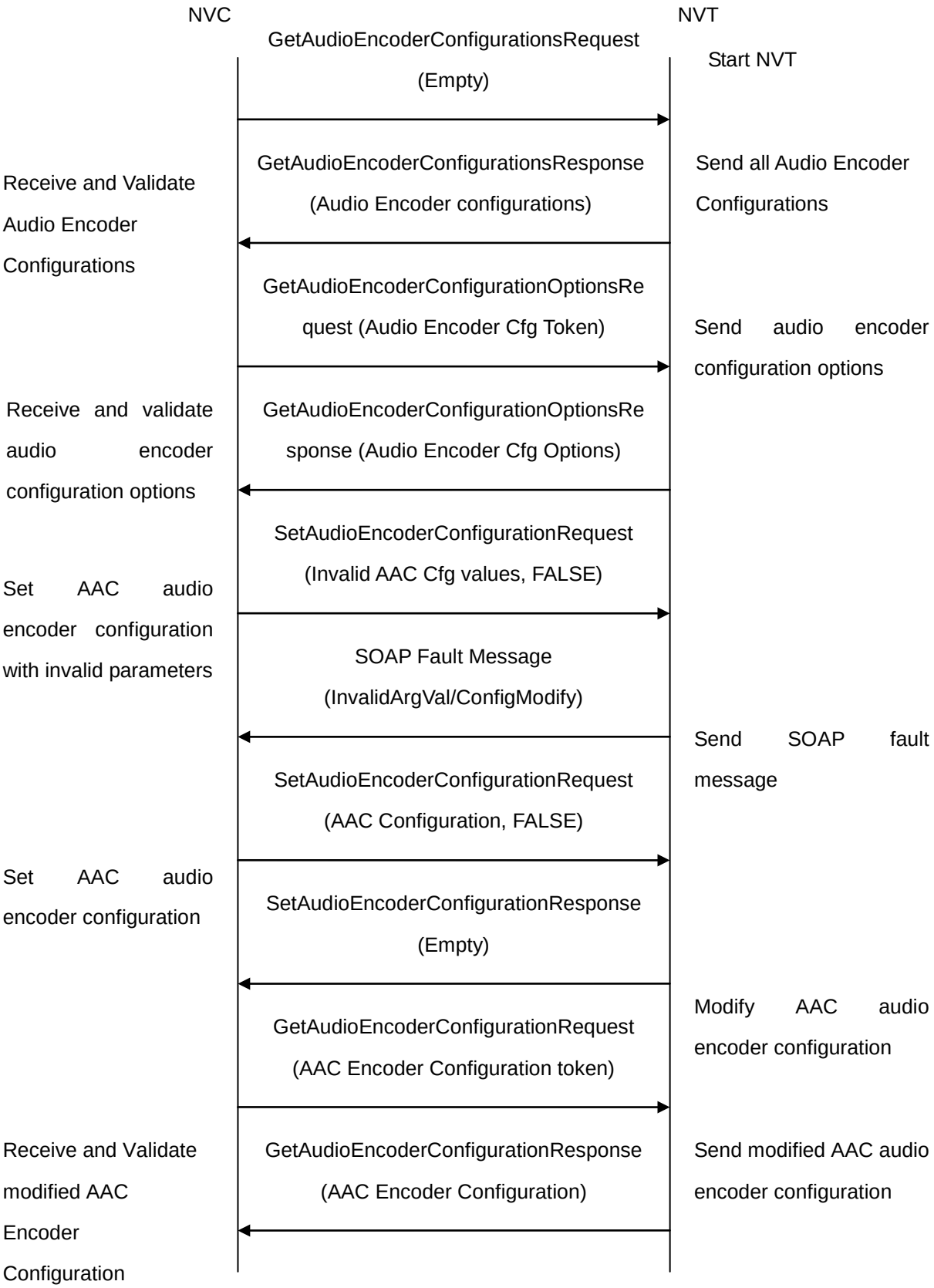
Requirement Level: MUST IF SUPPORTED (Audio) & IMPLEMENTED (AAC)

Test Purpose: To verify NVT AAC Audio Encoder Configuration Setting

Pre-Requisite: Audio is supported by NVT and AAC is implemented by NVT.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
GetAudioEncoderConfigurationsRequest
(Empty)
Start NVT
Receive and Validate Audio Encoder Configurations
GetAudioEncoderConfigurationsResponse
(Audio Encoder configurations)
Send all Audio Encoder Configurations
GetAudioEncoderConfigurationOptionsRequest (Audio Encoder Cfg Token)
Send audio encoder configuration options
Receive and validate audio encoder configuration options
GetAudioEncoderConfigurationOptionsResponse (Audio Encoder Cfg Options)
Set AAC audio encoder configuration with invalid parameters
SetAudioEncoderConfigurationRequest
(Invalid AAC Cfg values, FALSE)
SOAP Fault Message
(InvalidArgVal/ConfigModify)
Send SOAP fault message
SetAudioEncoderConfigurationRequest
(AAC Configuration, FALSE)
Set AAC audio encoder configuration
SetAudioEncoderConfigurationResponse
(Empty)
Modify AAC audio encoder configuration
GetAudioEncoderConfigurationRequest
(AAC Encoder Configuration token)
Receive and Validate modified AAC Encoder Configuration
GetAudioEncoderConfigurationResponse
(AAC Encoder Configuration)
Send modified AAC audio encoder configuration

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes GetAudioEncoderConfigurations request.
4. NVT sends the list of supported audio encoder configurations in GetAudioEncoderConfigurationsResponse message.
5. NVC invokes GetAudioEncoderConfigurationOptions Request (Audio Encoder Configuration token) to retrieve audio encoder configuration options for the specified audio encoder configuration.
6. NVT sends the range of configurable values for the received audio encoder configuration in the GetAudioEncoderConfigurationOptionsResponse message.
7. Test steps -5 & 6 have to be repeated for all audio encoder configurations until NVC finds a audio encoder configuration with AAC encoding support
8. NVC invokes SetAudioEncoderConfiguration request with AAC configuration values outside the range defined in the GetAudioEncoderConfigurationOptionsResponse and **'ForcePersistence'** flag as **'FALSE'**.
9. NVT send the SOAP 1.2 fault message **(InvalidArgVal/ConfigModify)**.
10. NVC verifies the SOAP fault message sent by NVT.
11. NVC invokes SetAudioEncoderConfiguration request (**Encoding = "AAC", Bit Rate = r1, Sample Rate = r2 and force persistence = false**). These values will be taken from the GetAudioEncoderConfigurationOptionsResponse message.
12. NVT modifies AAC audio encoder configuration and responds with SetAudioEncoderConfigurationResponse message indicating success.
13. NVC verifies the AAC audio Encoder Configuration settings on NVT by invoking GetAudioEncoderConfiguration request.
14. NVT sends modified AAC audio Encoder Configuration in the GetAudioEncoderConfigurationResponse message (**Encoding = "AAC", Bit Rate = r1, Sample Rate = r2**).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send GetAudioEncoderConfigurationsResponse message.

DUT did not send GetAudioEncoderConfigurationOptionsResponse message.

DUT doesn't support AAC audio encoding.

DUT did not send the SOAP 1.2 fault message (InvalidArgVal/ConfigModify) for invalid SetAudioEncoderConfiguration request.

DUT did not send SetAudioEncoderConfigurationResponse message.

DUT did not send GetAudioEncoderConfigurationResponse message.

The DUT did not modify AAC Audio Encoder Configuration.

7.4 PTZ Configuration

7.4.1 NVT PTZ CONFIGURATION

Test Label: Media Configuration NVT PTZ Configuration

ONVIF Core Specification Coverage: 10.2.2 Get media profiles, 10.2.8 Add PTZ configuration to a profile, 10.2.15 Remove PTZ configuration from a profile, 10.2.18 Delete media profile, 13.3.1 GetConfigurations.

Device Type: NVT

Command Under Test: None

WSDL Reference: media.wsdl

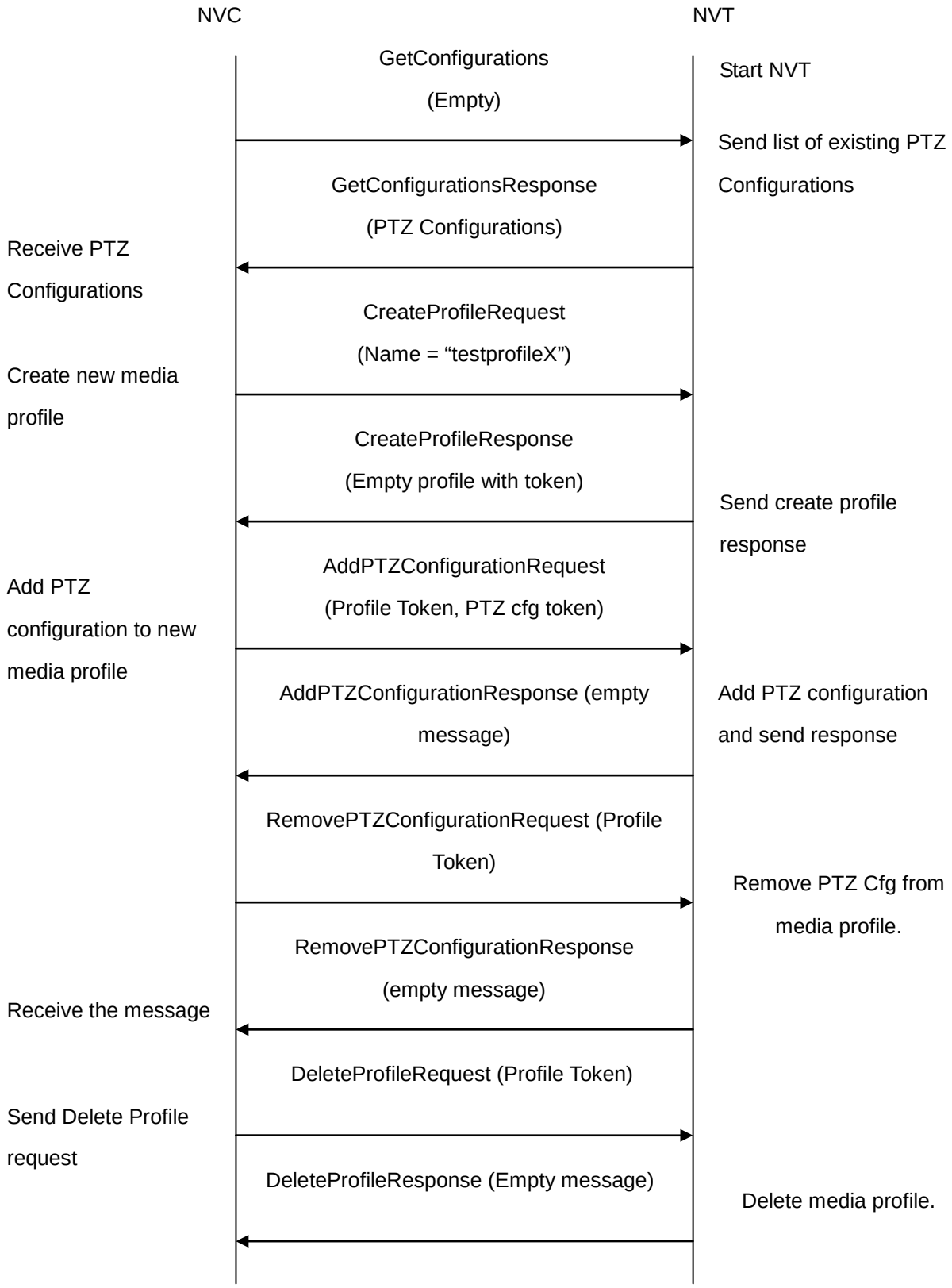
Requirement Level: MUST IF SUPPORTED (PTZ)

Test Purpose: To verify NVT PTZ Configuration Operations

Pre-Requisite: PTZ is supported by NVT

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
GetConfigurations
(Empty)
Start NVT
Send list of existing PTZ Configurations
GetConfigurationsResponse
(PTZ Configurations)
Receive PTZ Configurations
CreateProfileRequest
(Name = “testprofileX”)
Create new media profile
CreateProfileResponse
(Empty profile with token)
Send create profile response
AddPTZConfigurationRequest
(Profile Token, PTZ cfg token)
Add PTZ configuration to new media profile
AddPTZConfigurationResponse (empty message)
Add PTZ configuration and send response
RemovePTZConfigurationRequest (Profile Token)
Remove PTZ Cfg from media profile.
RemovePTZConfigurationResponse
(empty message)
Receive the message
DeleteProfileRequest (Profile Token)
Send Delete Profile request
DeleteProfileResponse (Empty message)
Delete media profile.

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes GetConfigurations request on PTZ service of NVT to retrieve the list of PTZ configurations supported by NVT.
4. NVC validates the GetConfigurationsResponse message sent by the NVT. At least one PTZ configuration should be present in the response.
5. NVC invokes CreateProfile request with **ProfileToken** = **'testprofileX'**.
6. NVT creates new media profile and sends the response.
7. NVC invokes AddPTZConfiguration request with **ProfileToken** as **'testprofileX'** and **ConfigurationToken** as one of the existing PTZConfiguration tokens.
8. NVT adds the PTZ configuration to the profile and sends the response.
9. NVC invokes RemovePTZConfiguration request with **ProfileToken** as **'testprofileX'**.
10. NVT removes the PTZ configuration from media profile and sends the response.
11. NVC invokes DeleteProfile request with **ProfileToken** as **'testprofileX'**.
12. NVT deletes the media profile and sends the response.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send valid GetConfigurationsResponse message.

DUT doesn't have PTZ configuration.

DUT did not send AddPTZConfigurationResponse message.

DUT did not send RemovePTZConfigurationResponse message.

DUT did not send DeleteProfileResponse message.

7.5 Metadata Configuration

7.5.1 NVT METADATA CONFIGURATION

Test Label: Media Configuration NVT Metadata Configuration

ONVIF Core Specification Coverage: 10.2.1 Create media profile, 10.2.10 Add metadata configuration to a profile, 10.2.17 Remove metadata configuration from a profile, 10.2.18 Delete media profile, 10.10.1 Get metadata configurations, 10.10.2 Get metadata configuration, 10.10.3 Get

compatible metadata configurations, 10.10.4 Get metadata configuration options, 10.10.5 Modify a metadata configuration.

Device Type: NVT

Command Under Test: None

WSDL Reference: media.wsdl

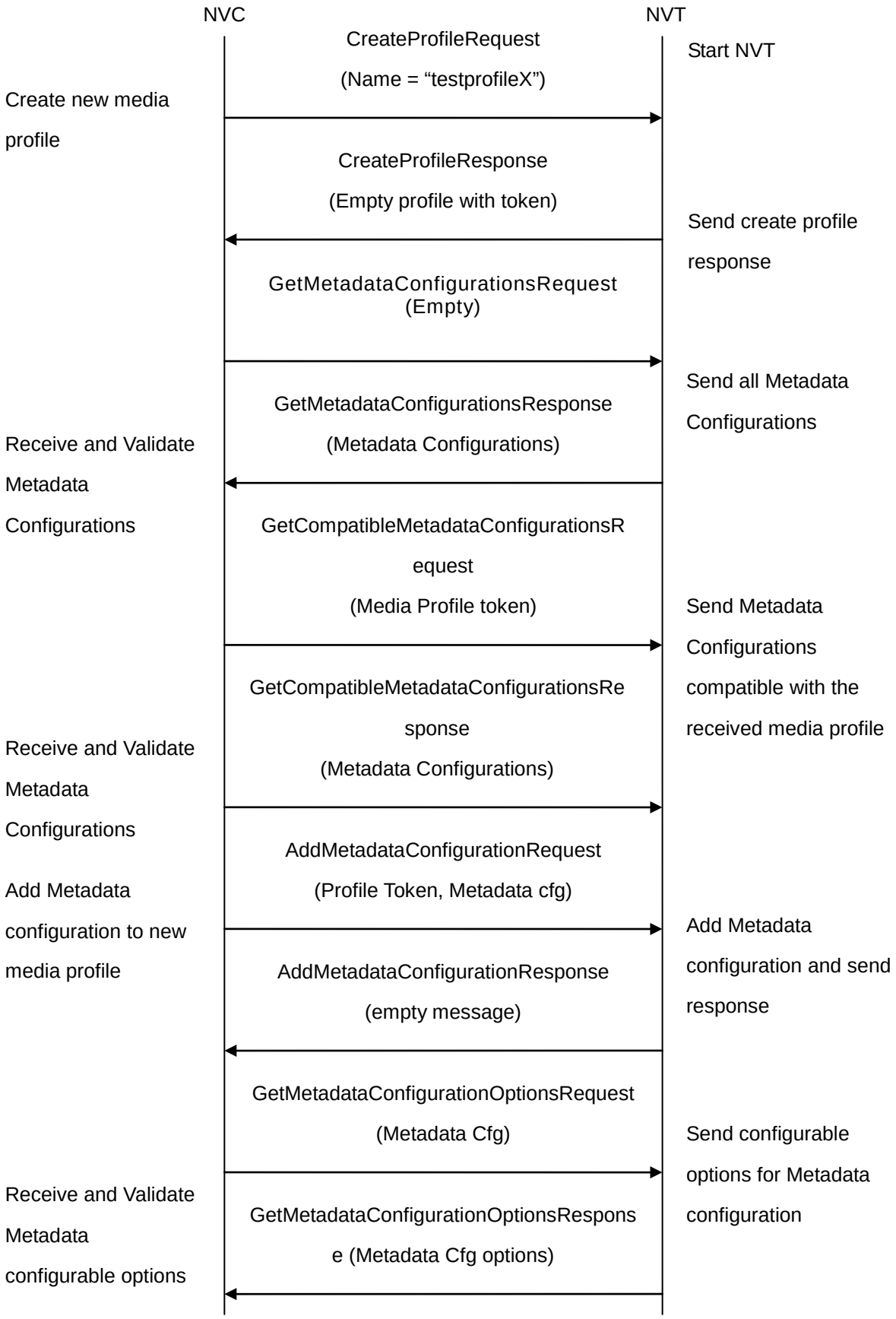
Requirement Level: MUST

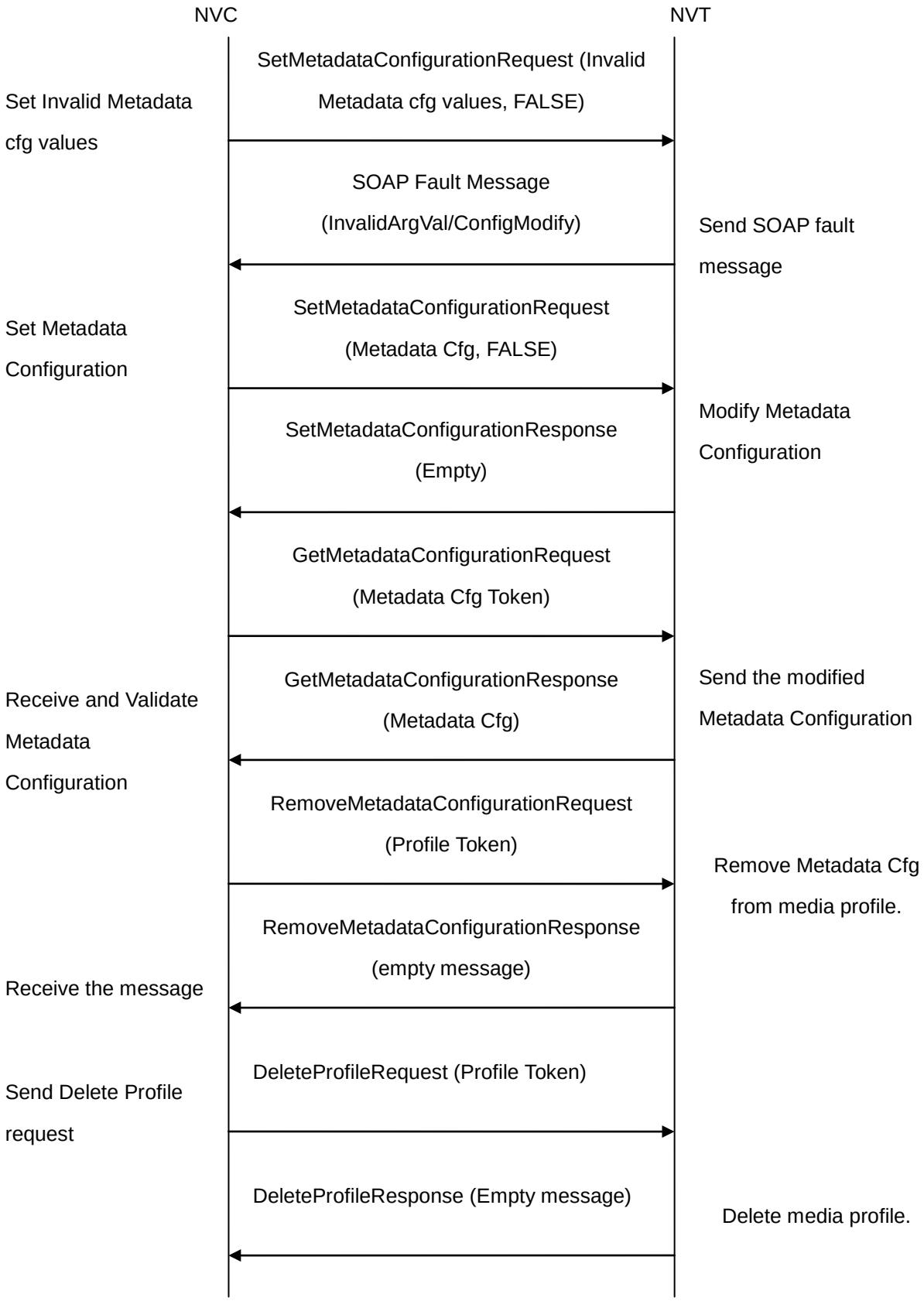
Test Purpose: To verify NVT Metadata Configuration Operations

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
CreateProfileRequest
(Name = “testprofileX”)
Start NVT
Create new media profile
CreateProfileResponse
(Empty profile with token)
Send create profile response
GetMetadataConfigurationsRequest
(Empty)
Send all Metadata Configurations
GetMetadataConfigurationsResponse
(Metadata Configurations)
Receive and Validate Metadata Configurations
GetCompatibleMetadataConfigurationsRequest
(Media Profile token)
Send Metadata Configurations compatible with the received media profile
GetCompatibleMetadataConfigurationsResponse
(Metadata Configurations)
Receive and Validate Metadata Configurations
AddMetadataConfigurationRequest
(Profile Token, Metadata cfg)
Add Metadata configuration to new media profile
Add Metadata configuration and send response
AddMetadataConfigurationResponse
(empty message)
GetMetadataConfigurationOptionsRequest
(Metadata Cfg)
Send configurable options for Metadata configuration
GetMetadataConfigurationOptionsResponse (Metadata Cfg options)
Receive and Validate Metadata configurable options


NVC
NVT
SetMetadataConfigurationRequest (Invalid Metadata cfg values, FALSE)
Set Invalid Metadata cfg values
SOAP Fault Message (InvalidArgVal/ConfigModify)
Send SOAP fault message
SetMetadataConfigurationRequest (Metadata Cfg, FALSE)
Set Metadata Configuration
Modify Metadata Configuration
SetMetadataConfigurationResponse (Empty)
GetMetadataConfigurationRequest (Metadata Cfg Token)
GetMetadataConfigurationResponse (Metadata Cfg)
Send the modified Metadata Configuration
Receive and Validate Metadata Configuration
RemoveMetadataConfigurationRequest (Profile Token)
Remove Metadata Cfg from media profile.
RemoveMetadataConfigurationResponse (empty message)
Receive the message
DeleteProfileRequest (Profile Token)
Send Delete Profile request
DeleteProfileResponse (Empty message)
Delete media profile.

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes CreateProfile request with **ProfileToken** = **'testprofileX'**.
4. NVT creates new media profile and sends the response.
5. NVC will invoke GetMetadataConfigurations request to retrieve the list of metadata configurations supported by the NVT.
6. NVC verifies the list of metadata configurations sent by the NVT.
7. NVC invokes GetCompatibleMetadataConfigurations request with **'testprofileX'** as **ProfileToken**.
8. NVT sends the list of metadata configurations compatible with the received media profile token.
9. NVC invokes AddMetadataConfiguration request message with **ProfileToken** as **'testprofileX'** and **ConfigurationToken** as one of the metadata configuration tokens received in the GetCompatibleMetadataConfigurations response.
10. NVT adds the Metadata configuration to the profile and sends the response.
11. NVC invokes GetMetadataConfigurationOptions request with **ConfigurationToken** as same token in AddMetadataConfiguration request.
12. NVT sends the configurable options supported for the received metadata configuration.
13. NVC invokes SetMetadataConfiguration request with metadata configuration values outside the range defined in GetMetadataConfigurationOptions response and **'ForcePersistence'** flag as **'FALSE'**.
14. NVT send the SOAP 1.2 fault message **(InvalidArgVal/ConfigModify)**.
15. NVC verifies the SOAP fault message sent by NVT.
16. NVC invokes SetMetadataConfiguration request with metadata configuration values within the range defined in GetMetadataConfigurationOptions response and **'ForcePersistance'** flag as **'FALSE'**.
17. NVT modifies the metadata configuration and sends the SetMetadataConfiguration indicating success.
18. NVC verifies the modified Metadata configuration by invoking the GetMetadataConfiguration request.
19. NVT sends the modified Metadata configuration in GetMetadataConfiguration response.
20. NVC invokes RemoveMetadataConfiguration request with **ProfileToken** as **'testprofileX'**.
21. NVT removes the Metadata configuration token from media profile and sends the response.
22. NVC invokes DeleteProfile request with **ProfileToken** as **'testprofileX'**.
23. NVT deletes the media profile and sends the response.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send CreateProfileResponse message.

DUT did not send GetMetadataConfigurationsResponse message.

DUT did not send GetCompatibleMetadataConfigurationsResponse message.

DUT did not send AddMetadataConfigurationResponse message.

DUT did not send GetMetadataConfigurationOptionsResponse message.

DUT did not send the SOAP 1.2 fault message (InvalidArgVal/ConfigModify) for invalid SetMetadataConfiguration request.

DUT did not send SetMetadataConfigurationResponse message.

DUT did not send GetMetadataConfigurationResponse message.

DUT did not modify metadata configuration correctly.

DUT did not send RemoveMetadataConfigurationResponse message.

DUT did not send DeleteProfileResponse message.

7.6 Media Streaming

7.6.1 NVT SNAPSHOT URI

Test Label: Media Configuration NVT Snapshot URI

ONVIF Core Specification Coverage: 10.2.2 Get media profiles, 10.12.1 Request snapshot URI.

Device Type: NVT

Command Under Test: GetSnapshotUri

WSDL Reference: media.wsdl

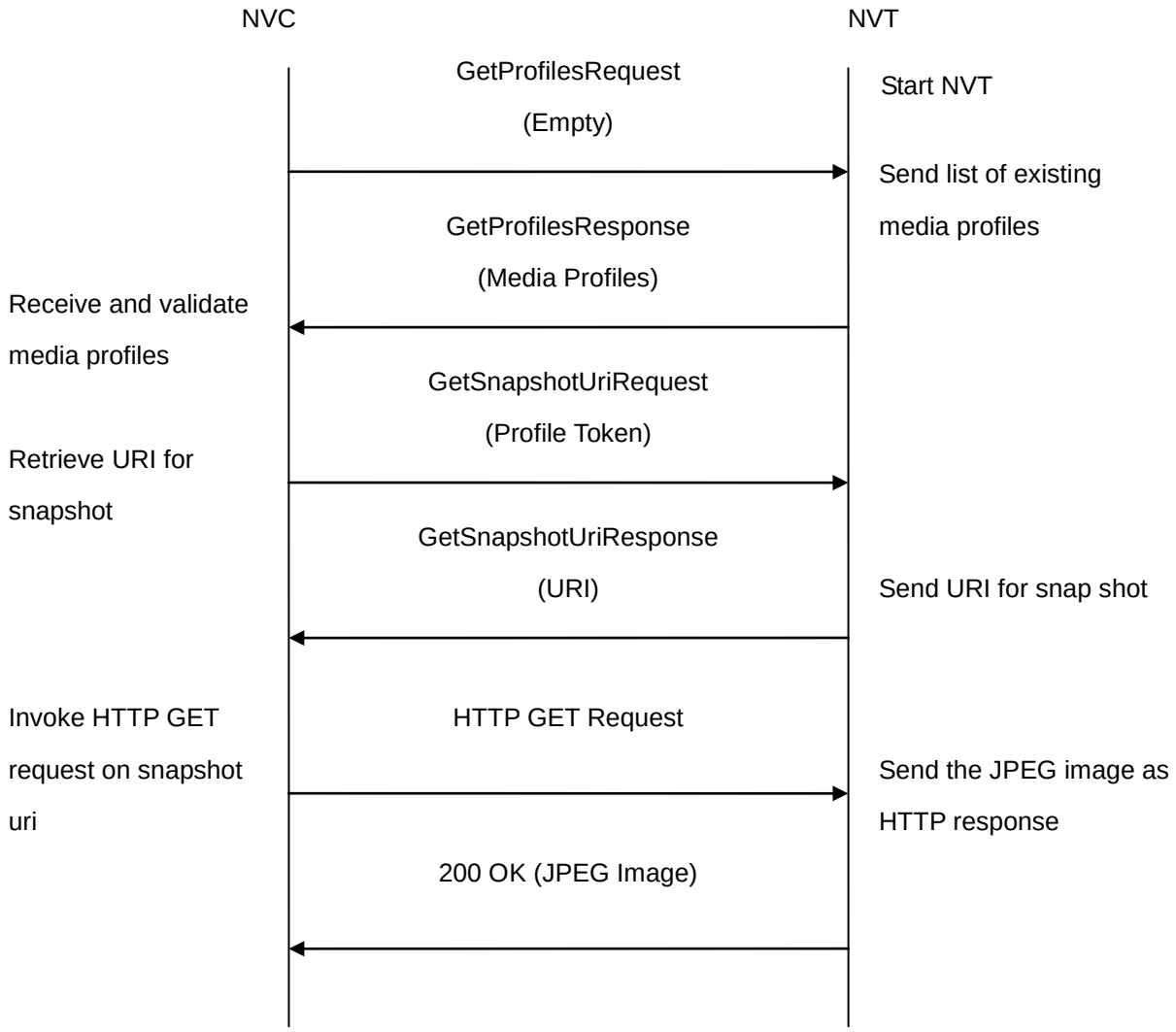
Requirement Level: SHOULD

Test Purpose: To retrieve snapshot URI of NVT for given media profile

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes GetProfiles request to retrieve the list of existing media profiles on NVT.
4. NVC validates the GetProfilesResponse message sent by the NVT. At least one media profile should be present with video source and video encoder.
5. NVC invokes GetSnapshotUri request with **ProfileToken** as one of the media profile tokens received in GetProfilesResponse message.
6. NVT sends HTTP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetSnapshotUriResponse message.
7. NVC invokes HTTP GET request on the snapshot URI sent by NVT.
8. NVT sends 200 OK message and the single shot JPEG image data.

9. NVC verifies the JPEG image sent by the NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send GetProfilesResponse message.

DUT did not send GetSnapshotUriResponse message.

DUT did not send one or more mandatory parameters in the GetSnapshotUriResponse message (mandatory parameters –Uri, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send 200 OK message.

DUT did not send valid JPEG image data.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

7.7 Error Handling

7.7.1 NVT SOAP FAULT MESSAGE

Test Label: Media Configuration NVT generates SOAP 1.2 fault message for Invalid GetStreamUriRequest Message.

ONVIF Core Specification Coverage: 10.11.1 Request stream URI.

Device Type: NVT

Command Under Test: GetStreamUri

WSDL Reference: media.wsdl

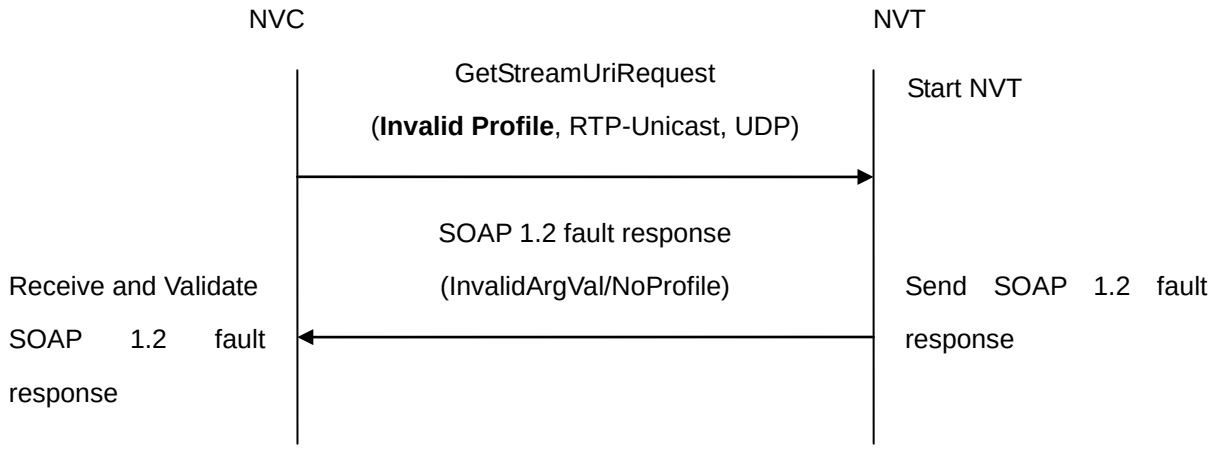
Requirement Level: SHOULD

Test Purpose: To verify that NVT generates SOAP 1.2 fault message to the invalid GetStreamUriRequest message (Invalid Media Profile).

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetStreamUriRequest message with **invalid media profile**.
4. NVT will generate the SOAP 1.2 fault message **(InvalidArgVal/NoProfile)**.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct SOAP 1.2 fault message (fault code, namespace etc).

Note: See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

7.7.2 NVT SOAP FAULT MESSAGE

Test Label: Media Configuration NVT generates SOAP 1.2 fault message for Invalid GetStreamUriRequest Message.

ONVIF Core Specification Coverage: 10.11.1 Request stream URI.

Device Type: NVT

Command Under Test: GetStreamUri

WSDL Reference: media.wsdl

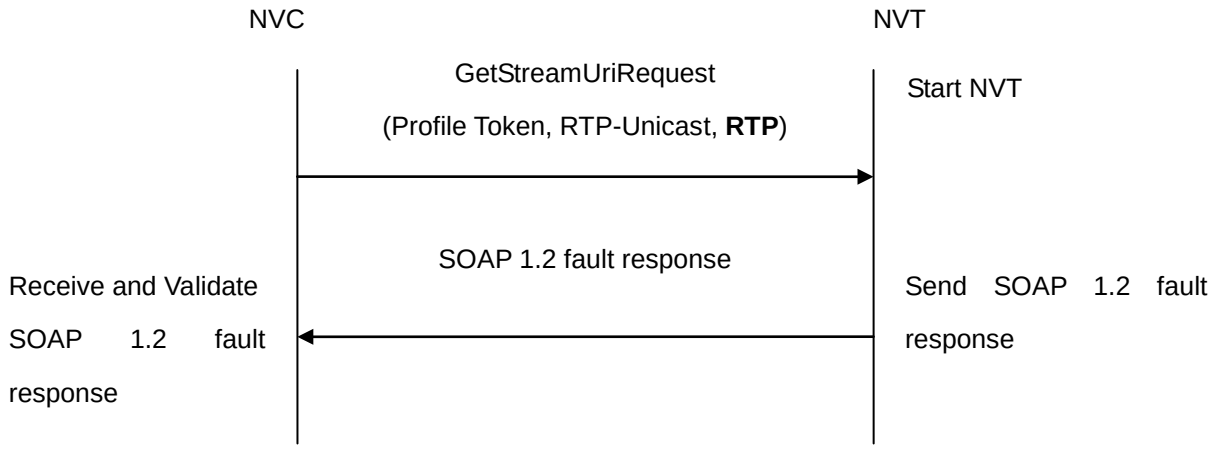
Requirement Level: SHOULD

Test Purpose: To verify that NVT generates SOAP 1.2 fault message to the invalid GetStreamUriRequest message (Invalid Transport).

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetStreamUriRequest message with invalid Transport (**RTP**).
4. NVT will generate the SOAP 1.2 fault message.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct SOAP 1.2 fault message (fault code, namespace etc).

Note: See Annex A.7 for correct syntax for the StreamSetup element in GetStreamUri requests.

8 Real Time Streaming Test Cases

8.1 Video Streaming

8.1.1 NVT MEDIA CONTROL – RTSP/TCP

Test Label: Real Time Viewing NVT RTSP control messages.

ONVIF Core Specification Coverage: 11.2.1 Stream control, 11.2.1.1 RTSP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

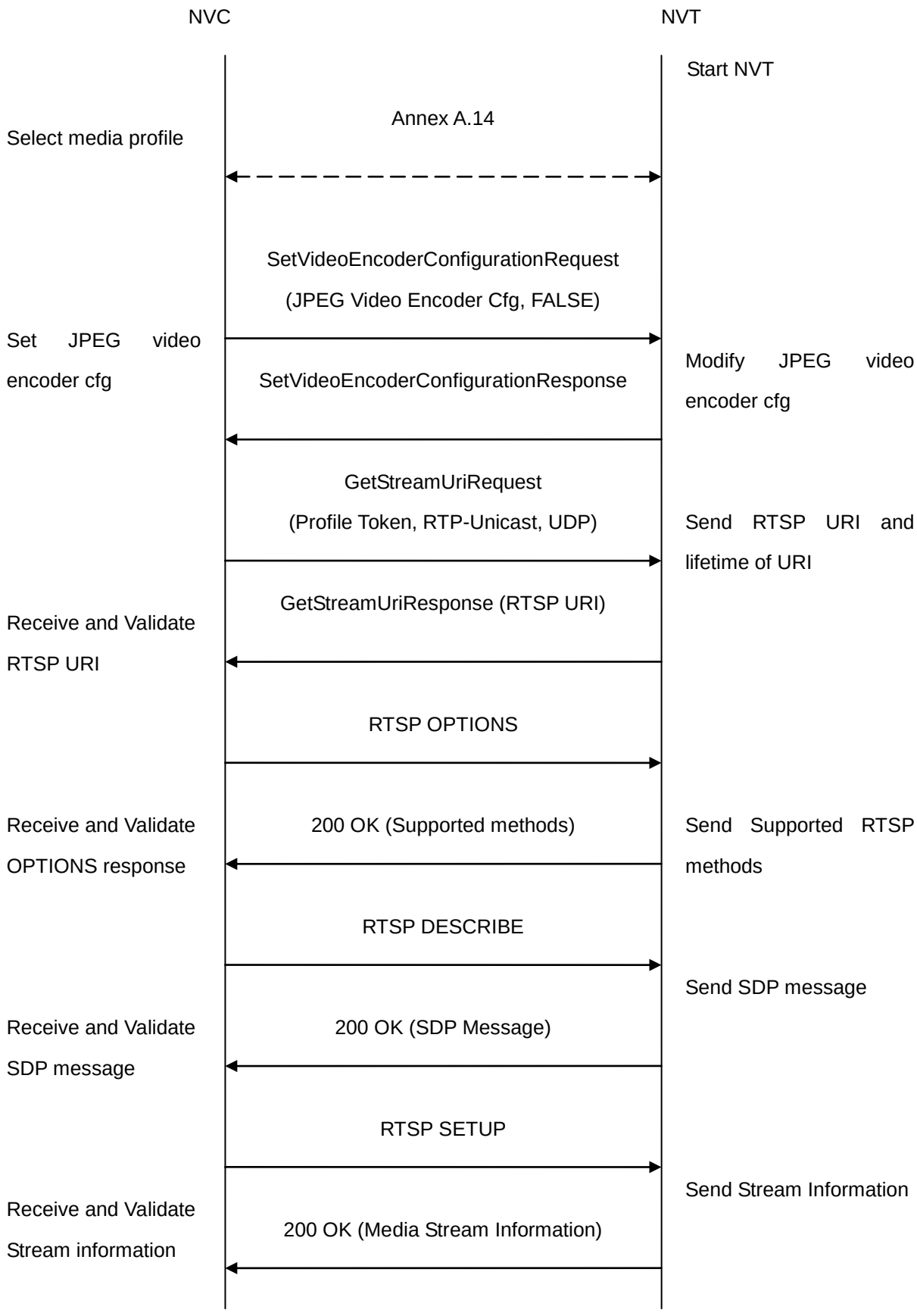
Requirement Level: MUST

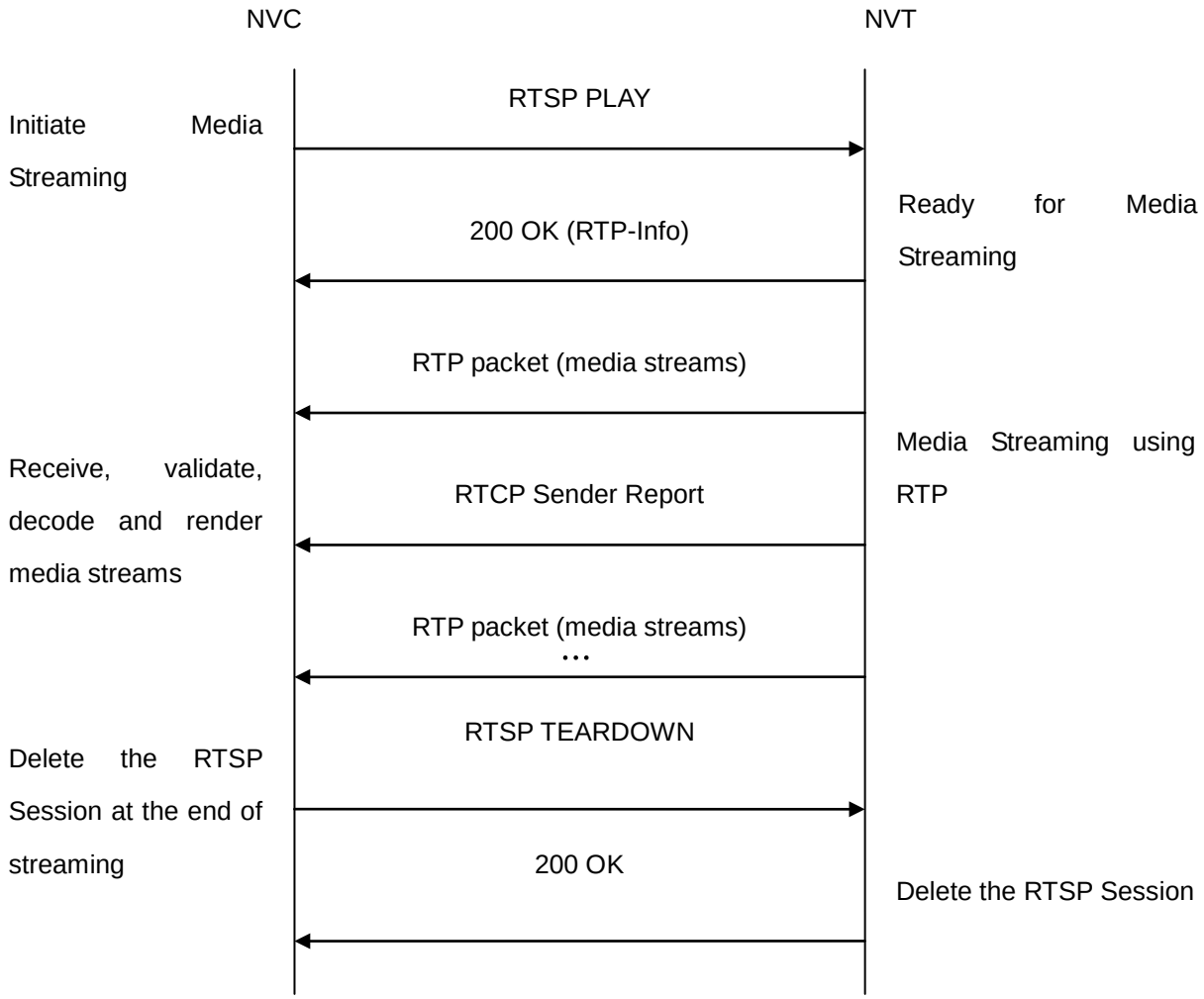
Test Purpose: To verify RTSP control messages of NVT.

Pre-Requisite: A media profile with JPEG video encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.14
Select media profile
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
Modify JPEG video encoder cfg
GetStreamUriRequest
(Profile Token, RTP-Unicast, UDP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP OPTIONS
Receive and Validate OPTIONS response
200 OK (Supported methods)
Send Supported RTSP methods
RTSP DESCRIBE
Send SDP message
Receive and Validate SDP message
200 OK (SDP Message)
RTSP SETUP
Send Stream Information
Receive and Validate Stream information
200 OK (Media Stream Information)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with JPEG video encoding support by following the procedure mentioned in Annex A.14.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.14.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, UDP transport**) to retrieve media stream URI for the selected media profile.

7. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
8. NVC verifies the RTSP media stream URI provided by the NVT.
9. NVC will invoke RTSP OPTIONS control request to understand the RTSP methods supported by NVT.
10. NVT sends 200 OK Response and list of supported RTSP methods.
11. NVC will invoke RTSP DESCRIBE control request to retrieve the media description information.
12. NVT sends 200 OK Response and SDP message.
13. NVC validates the session description information in the SDP message.
14. NVC will invoke RTSP SETUP control request to create a RTSP Session.
15. NVT sends 200 OK Response and Stream Information details.
16. NVC Verifies “Transport”, “Session” and “timeout” header fields in the SETUP response message.
17. NVC will invoke RTSP PLAY control request to initiate the media streaming.
18. NVT sends 200 OK Response and RTP protocol information.
19. NVC verifies “Session”, “RTP-Info”, “seq”, “uri” and “rtptime” header fields in the PLAY response message.
20. NVT transfers media streams over RTP/UDP.
21. NVT sends RTCP sender report to NVC.
22. NVC validates RTCP packets.
23. NVC validates RTP header for each media stream and render it after the validation.
24. NVC will invoke RTSP TEARDOWN control request to terminate the RTSP session at the end of the streaming.
25. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send mandatory RTSP commands DESCRIBE, SETUP, PLAY, TEARDOWN and SET_PARAMETER in OPTIONS response.

DUT did not send correct media stream information in the SDP message.

DUT did not send mandatory headers or fields in the SETUP response message.

DUT did not send mandatory headers or fields in the PLAY response message.

DUT did not send RTSP 200 OK response for RTSP OPTIONS, DESCRIBE, SETUP and PLAY requests.

RTSP Session is terminated by DUT during media streaming.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.5 for Invalid RTP header definition.

8.1.2 NVT MEDIA STREAMING – RTSP KEEPALIVE

Test Label: Real Time Viewing NVT RTSP Keep-alive.

ONVIF Core Specification Coverage: 11.2.1.1.1 Keep-alive method for RTSP session.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

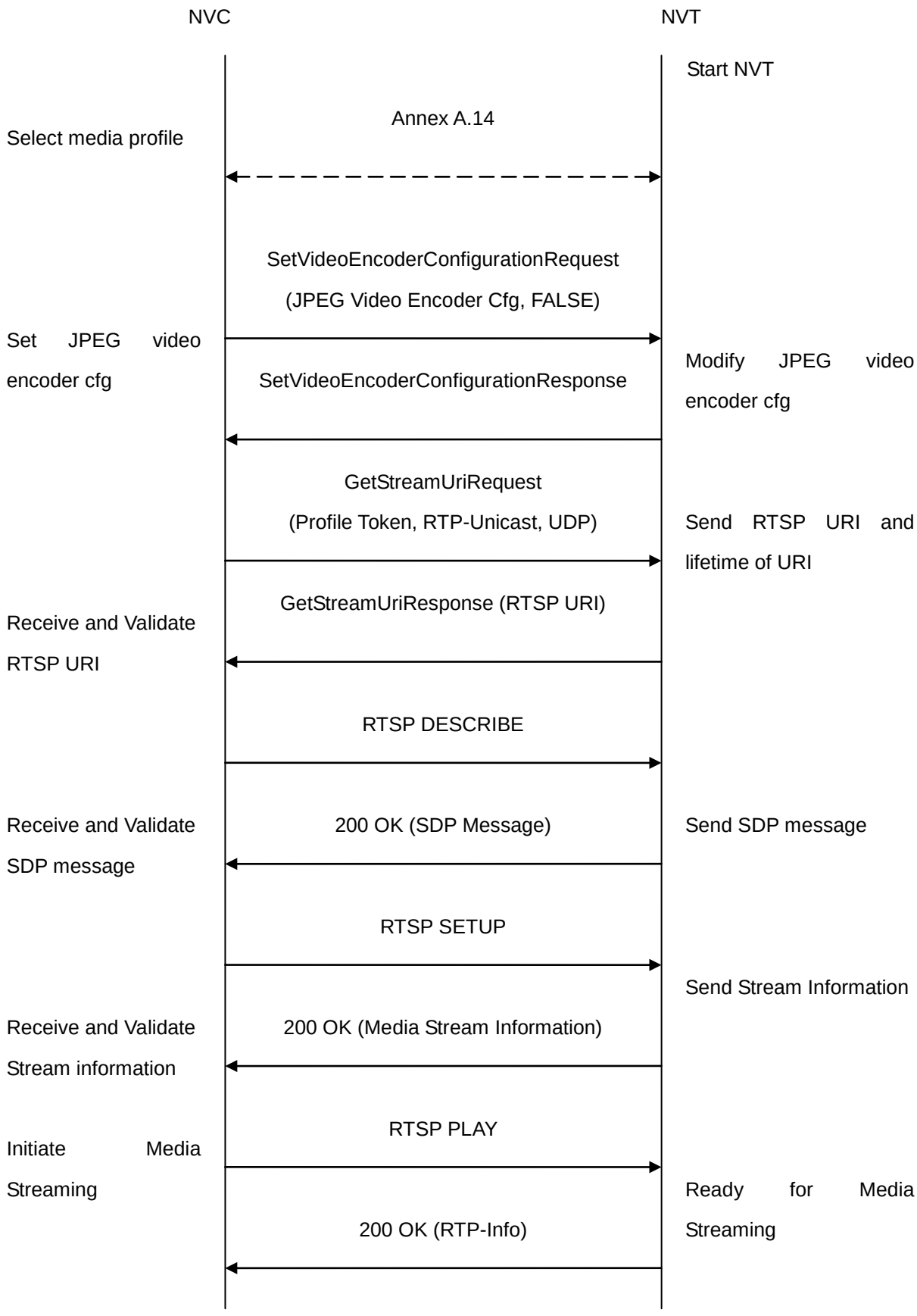
Requirement Level: MUST

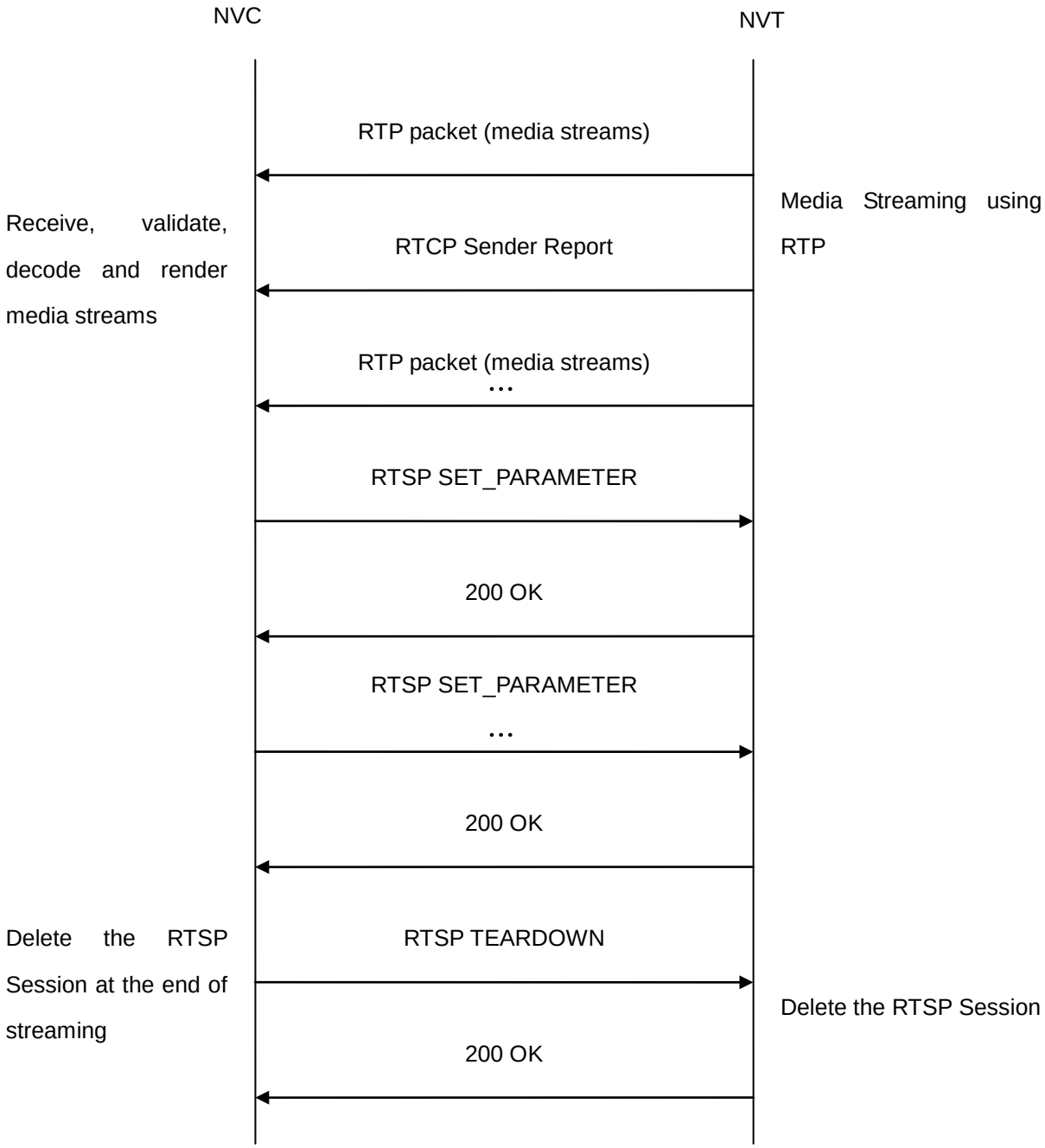
Test Purpose: To verify NVC and NVT exchange SET_PARAMETER messages during an active streaming session.

Pre-Requisite: A media profile with JPEG video encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.14
Select media profile
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
Modify JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
GetStreamUriRequest
(Profile Token, RTP-Unicast, UDP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Receive and Validate SDP message
200 OK (SDP Message)
Send SDP message
RTSP SETUP
Send Stream Information
Receive and Validate Stream information
200 OK (Media Stream Information)
RTSP PLAY
Initiate Media Streaming
Ready for Media Streaming
200 OK (RTP-Info)



Test Procedure:

1. Start an NVC.
2. Start an NVT.

3. NVC selects a media profile with JPEG video encoding support by following the procedure mentioned in Annex A.14.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.14.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, UDP transport**) to retrieve media stream URI for the selected media profile.
7. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
8. NVC verifies the RTSP media stream URI provided by the NVT.
9. NVC will invoke RTSP control requests (**DESCRIBE, SETUP and PLAY**).
10. NVC will verify "**Timeout**" header in the SETUP Response from NVT.
11. Based on the "**Timeout**" value, NVC will invoke **RTSP SET_PARAMETER** messages.
12. NVT will respond with 200 OK for RTSP SET_PARAMETER request.
13. Verify that the NVC and NVT are exchanging periodic SET_PARAMETER messages while a stream is being delivered.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send Timeout header in RTSP SETUP RESPONSE.

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and SET_PARAMETER requests.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

8.1.3 NVT MEDIA STREAMING – JPEG (RTP-Unicast/ UDP)

Test Label: Real Time Viewing NVT JPEG media streaming using RTP-Unicast/UDP transport.

ONVIF Core Specification Coverage: 11.1.1.1 RTP data transfer via UDP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

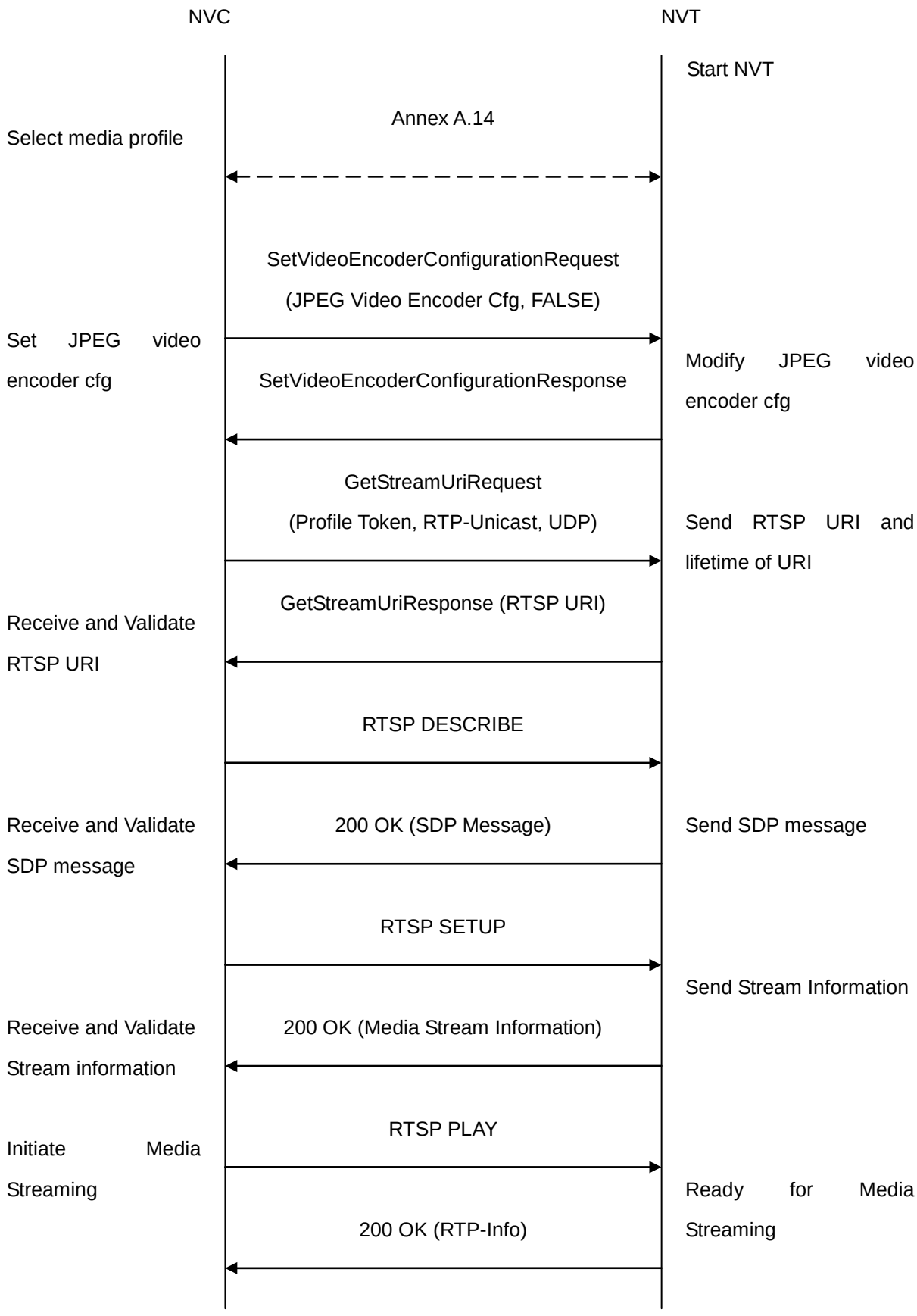
Requirement Level: MUST

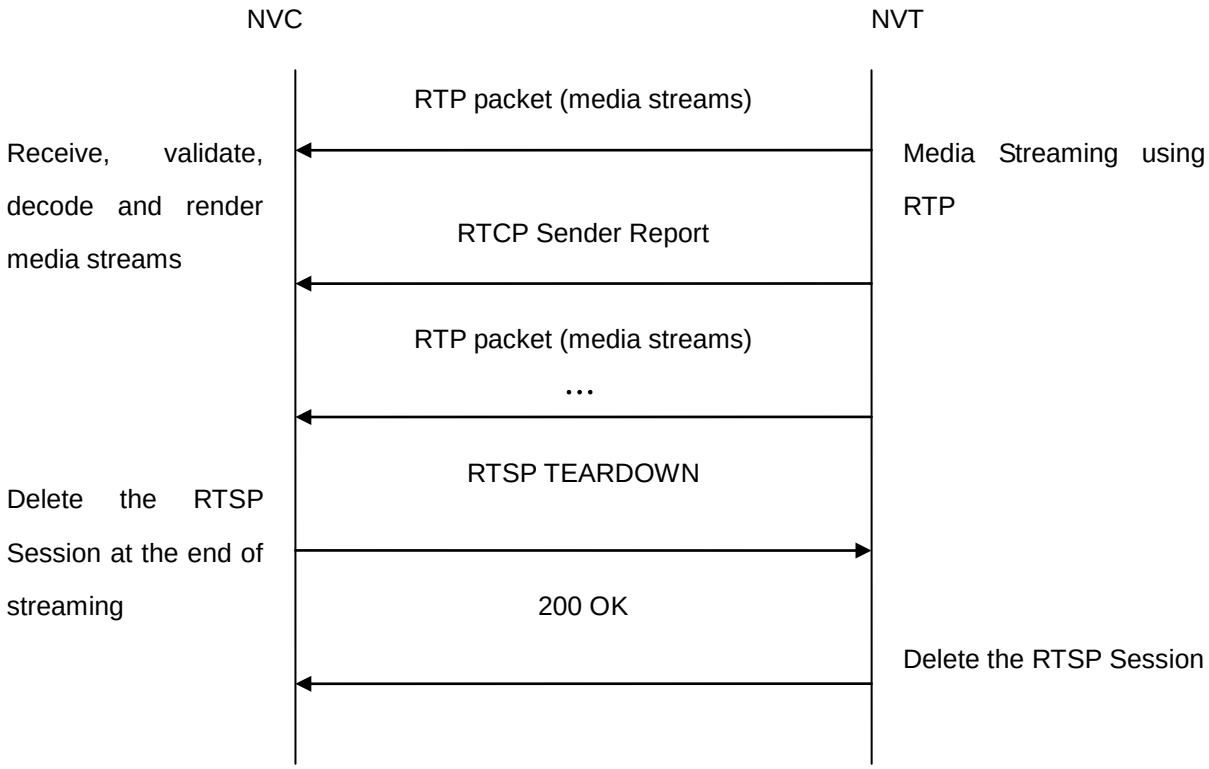
Test Purpose: To verify JPEG media streaming based on RTP/UDP Unicast Transport.

Pre-Requisite: A media profile with JPEG video encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.14
Select media profile
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
Modify JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
GetStreamUriRequest
(Profile Token, RTP-Unicast, UDP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Receive and Validate SDP message
200 OK (SDP Message)
Send SDP message
RTSP SETUP
Send Stream Information
Receive and Validate Stream information
200 OK (Media Stream Information)
RTSP PLAY
Initiate Media Streaming
Ready for Media Streaming
200 OK (RTP-Info)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with JPEG video encoding support by following the procedure mentioned in Annex A.14.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.14.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, UDP transport**) to retrieve media stream URI for the selected media profile.
7. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
8. NVC verifies the RTSP media stream URI provided by the NVT.
9. NVC invokes RTSP DESCRIBE request.
10. NVT sends 200 OK message and SDP information.
11. NVC invokes RTSP SETUP request with transport parameter as **RTP/UDP**.

12. NVT sends 200 OK message and the media stream information.
13. NVC invokes RTSP PLAY request.
14. NVT sends 200 OK message and starts media streaming.
15. NVT sends JPEG RTP media stream to NVC over UDP.
16. NVT sends RTCP sender report to NVC.
17. NVT validates the received RTP and RTCP packets, decodes and renders them.
18. NVC invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.
19. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.5 for Invalid RTP header definition.

8.1.4 NVT MEDIA STREAMING – JPEG (RTP-Unicast/RTSP/HTTP/TCP)

Test Label: Real Time Viewing NVT JPEG media streaming using HTTP transport.

ONVIF Core Specification Coverage: 11.1.1.4 RTP/RTSP/HTTP/TCP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP, 11.2.1.2 RTSP over HTTP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

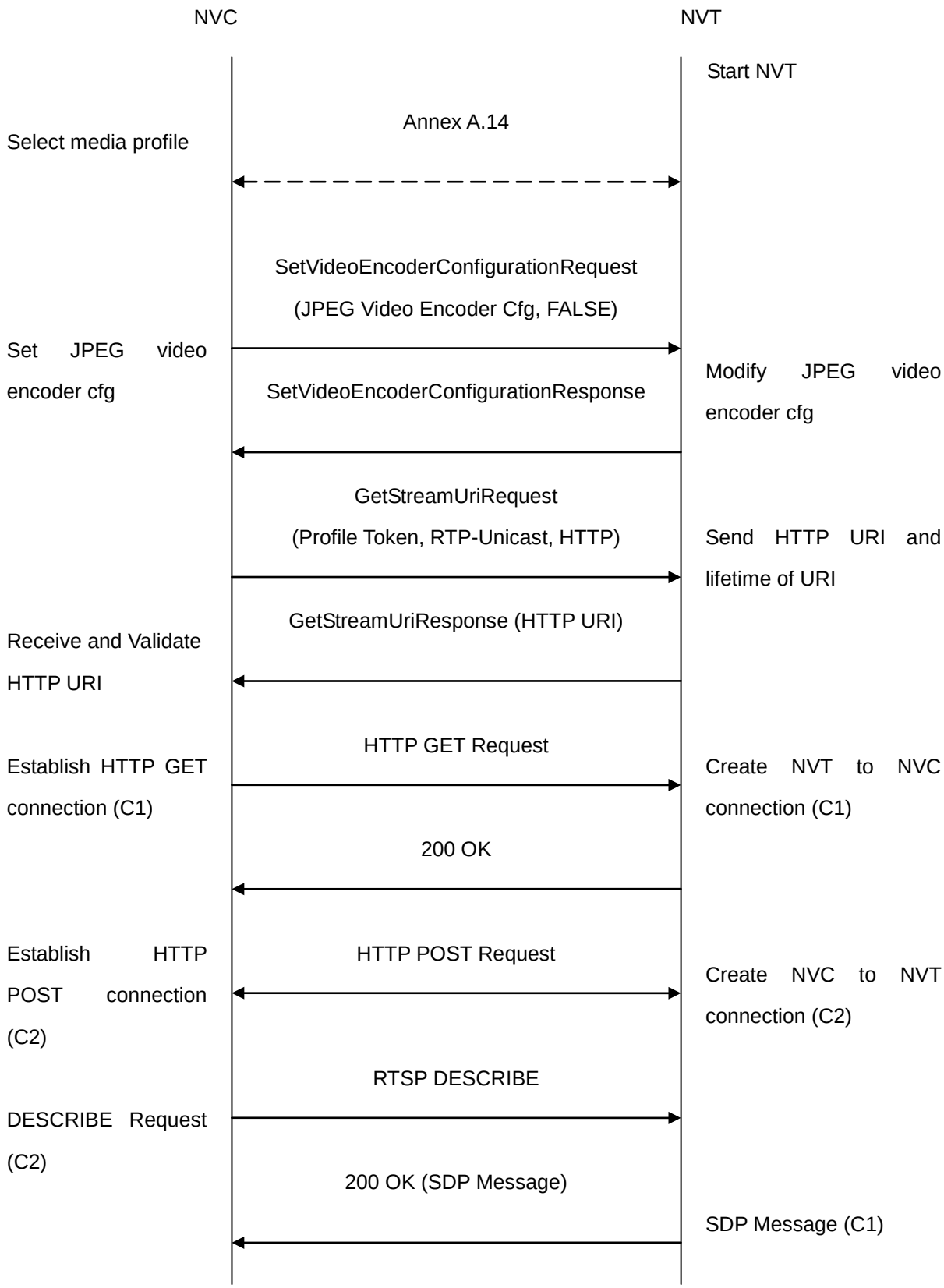
Requirement Level: MUST

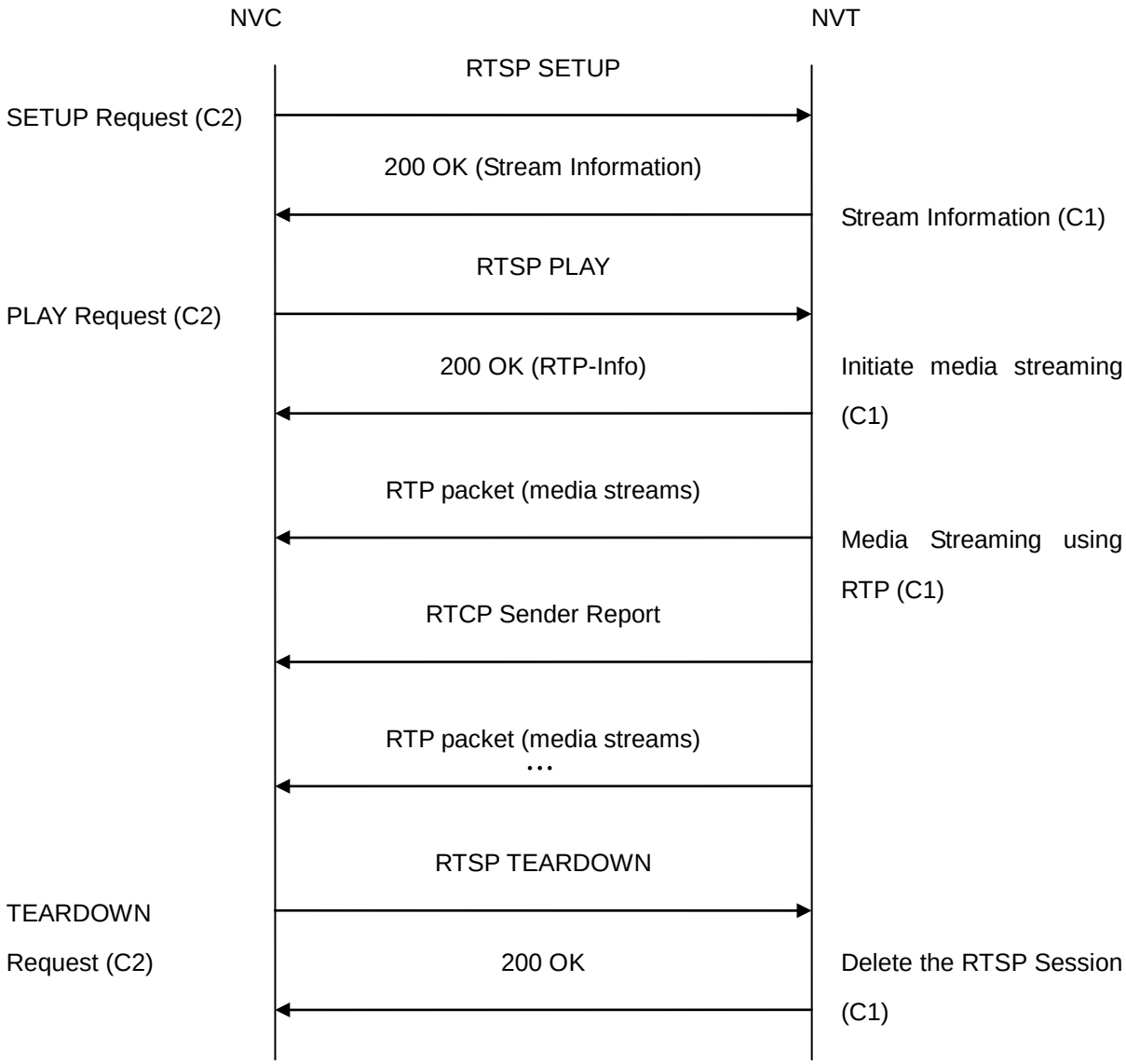
Test Purpose: To verify JPEG media streaming based on HTTP Transport.

Pre-Requisite: A media profile with JPEG video encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.14
Select media profile
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
Modify JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
GetStreamUriRequest
(Profile Token, RTP-Unicast, HTTP)
Send HTTP URI and lifetime of URI
GetStreamUriResponse (HTTP URI)
Receive and Validate HTTP URI
HTTP GET Request
Establish HTTP GET connection (C1)
Create NVT to NVC connection (C1)
200 OK
Establish HTTP POST connection (C2)
HTTP POST Request
Create NVC to NVT connection (C2)
RTSP DESCRIBE
DESCRIBE Request (C2)
200 OK (SDP Message)
SDP Message (C1)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with JPEG video encoding support by following the procedure mentioned in Annex A.14.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.14.

5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, HTTP transport**) to retrieve media stream URI for the selected media profile.
7. NVT sends HTTP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
8. NVC verifies the HTTP media stream URI provided by the NVT.
9. NVC invokes HTTP GET Request on NVT and establishes NVT to NVC connection for RTP data transfer.
10. NVC invokes HTTP POST Request and establishes NVC to NVT connection for RTSP control requests.
11. NVC invokes RTSP DESCRIBE request on HTTP POST connection.
12. NVT sends 200 OK message and SDP information on HTTP GET connection.
13. NVC invokes RTSP SETUP request on HTTP POST connection with transport parameter as **'RTP/TCP'** along with **'interleaved'** parameter.
14. NVT sends 200 OK message and the media stream information on HTTP GET connection.
15. NVC invokes RTSP PLAY request on HTTP POST connection.
16. NVT sends 200 OK message and starts media streaming on HTTP GET connection.
17. NVT transfers JPEG RTP media stream to NVC on HTTP GET connection.
18. NVT sends RTCP sender report to NVC on HTTP GET connection.
19. NVT validates the received RTP and RTCP packets, decodes and renders them.
20. NVC invokes RTSP TEARDOWN control request on HTTP POST connection and closes the HTTP POST connection.
21. NVT sends 200 OK Response on HTTP GET connection and closes the HTTP GET connection.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – HTTP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.5 for Invalid RTP header definition.

8.1.5 NVT MEDIA STREAMING – JPEG (RTP/RTSP/TCP)

Test Label: Real Time Viewing NVT JPEG media streaming using RTP/RTSP/TCP transport.

ONVIF Core Specification Coverage: 11.1.1.3 RTP/RTSP/TCP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF IMPLEMENTED (RTP/RTSP/TCP)

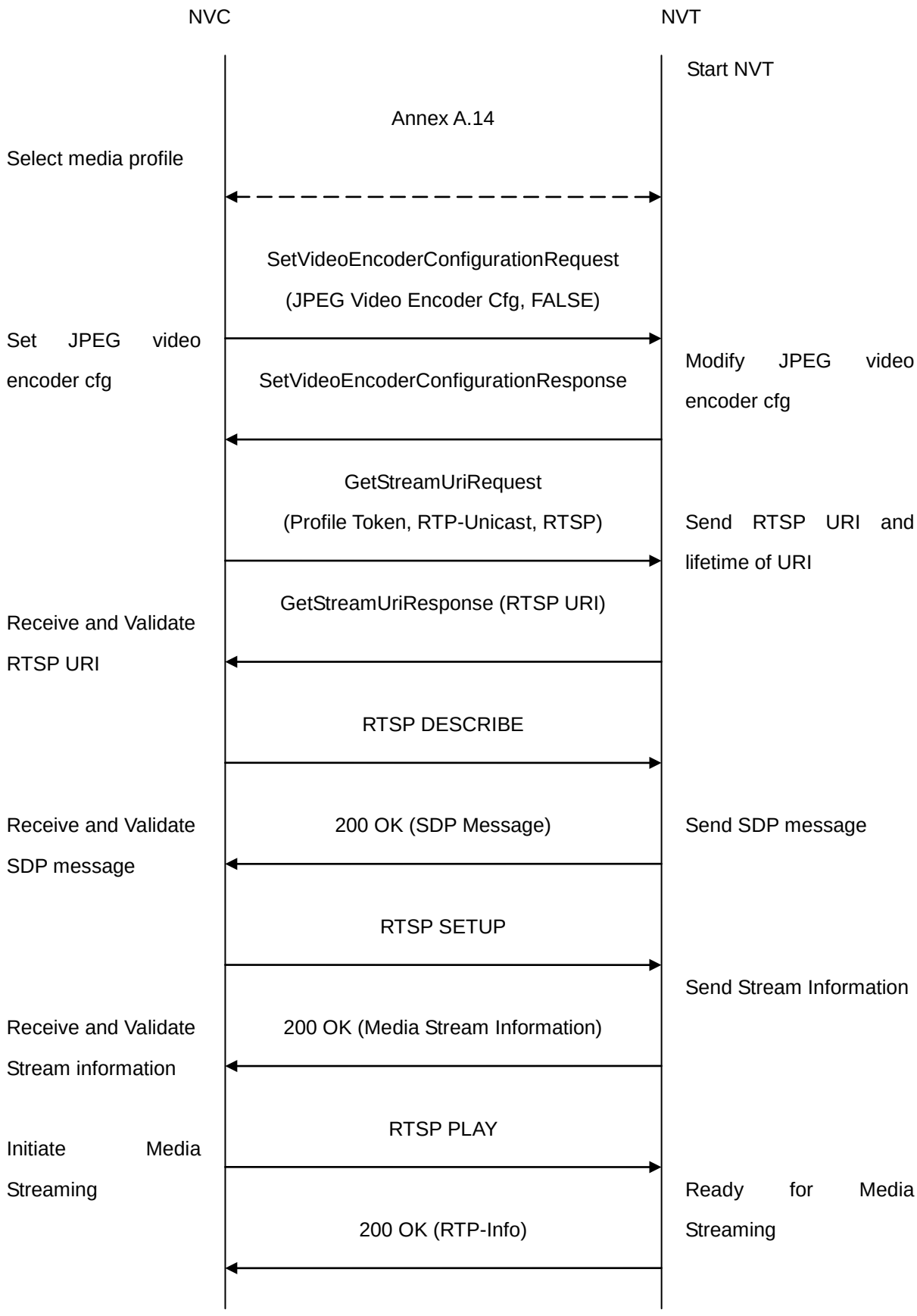
Test Purpose: To verify JPEG media streaming based on RTP/RTSP/TCP using RTSP tunnel.

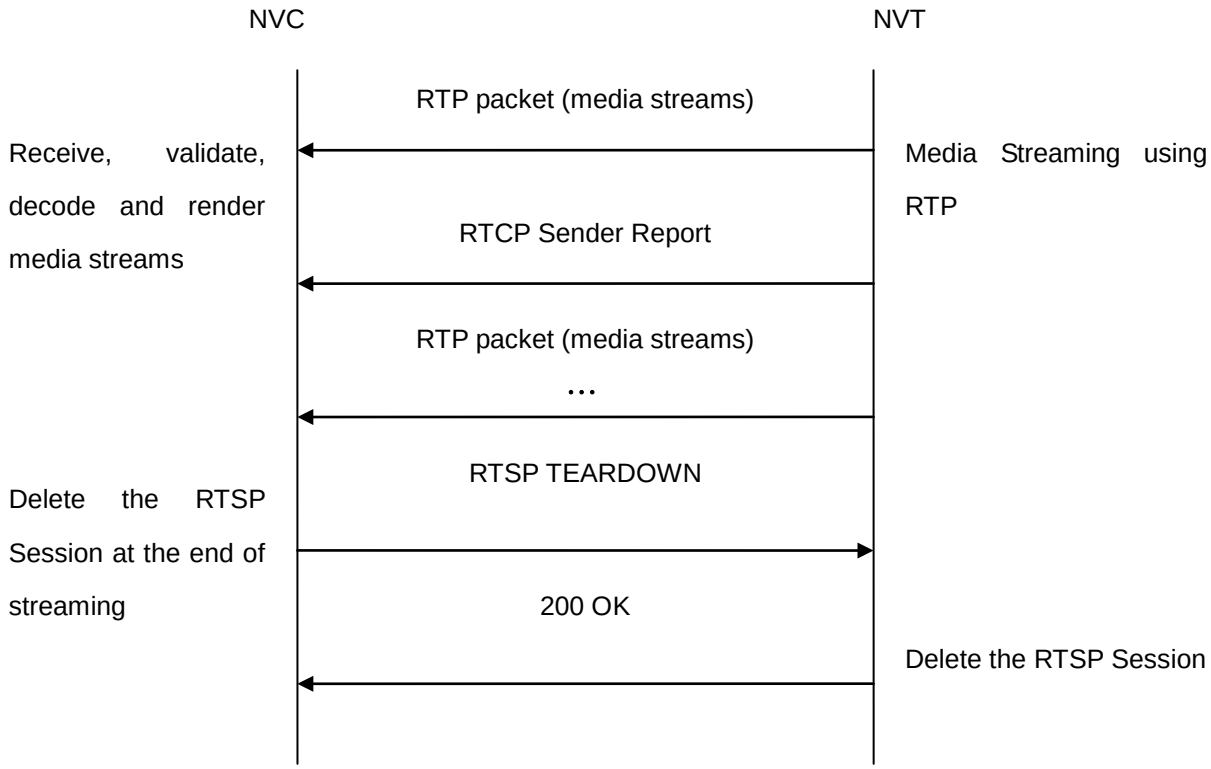
Pre-Requisite: RTP/RTSP/TCP media streaming is implemented by NVT.

A media profile with JPEG video encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.14
Select media profile
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
Modify JPEG video encoder cfg
GetStreamUriRequest
(Profile Token, RTP-Unicast, RTSP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Receive and Validate SDP message
200 OK (SDP Message)
Send SDP message
RTSP SETUP
Send Stream Information
Receive and Validate Stream information
200 OK (Media Stream Information)
RTSP PLAY
Initiate Media Streaming
Ready for Media Streaming
200 OK (RTP-Info)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with JPEG video encoding support by following the procedure mentioned in Annex A.14.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.14.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, RTSP transport**) to retrieve media stream URI for the selected media profile.
7. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
8. NVC verifies the RTSP media stream URI provided by the NVT.
9. NVC invokes RTSP DESCRIBE request.
10. NVT sends 200 OK message and SDP information.

11. NVC invokes RTSP SETUP request with transport parameter as **'RTP/TCP'** along with **'interleaved'** parameter.
12. NVT sends 200 OK message and the media stream information.
13. NVC invokes RTSP PLAY request.
14. NVT sends 200 OK message and starts media streaming.
15. NVT interleaves RTP and RTCP packets, send them over RTSP control connection.
16. NVT validates the received RTP and RTCP packets, decodes and renders them.
17. NVC invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.
18. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send RTP and RTCP packets as per [RFC 2326] section 10.12.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

8.1.6 NVT MEDIA STREAMING – MPEG4 (RTP-Unicast/ UDP)

Test Label: Real Time Viewing NVT MPEG4 media streaming using RTP-Unicast/UDP transport.

ONVIF Core Specification Coverage: 11.1.1.1 RTP data transfer via UDP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF IMPLEMENTED (MPEG4-SP)

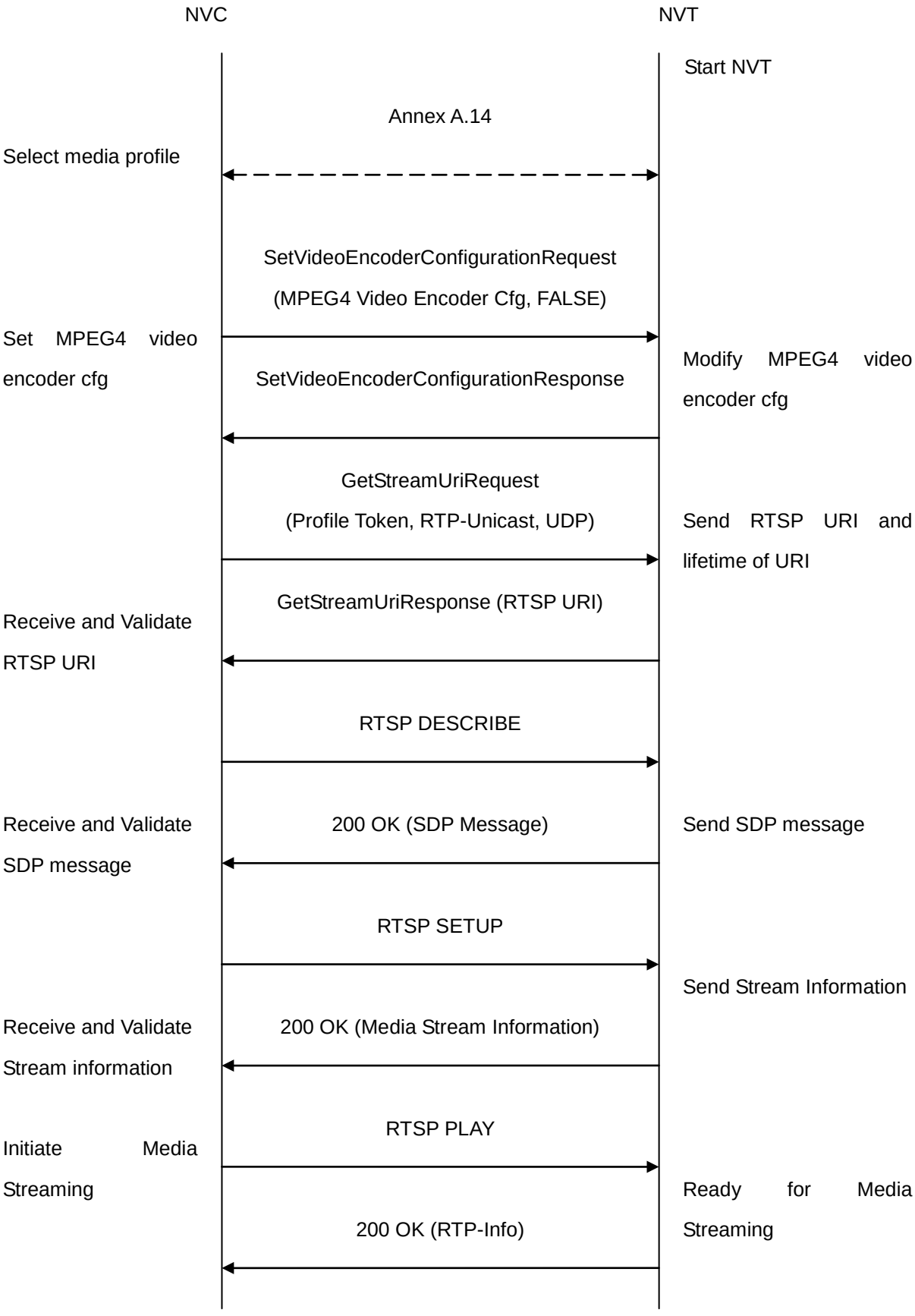
Test Purpose: To verify MPEG4 media streaming based on RTP/UDP Unicast Transport.

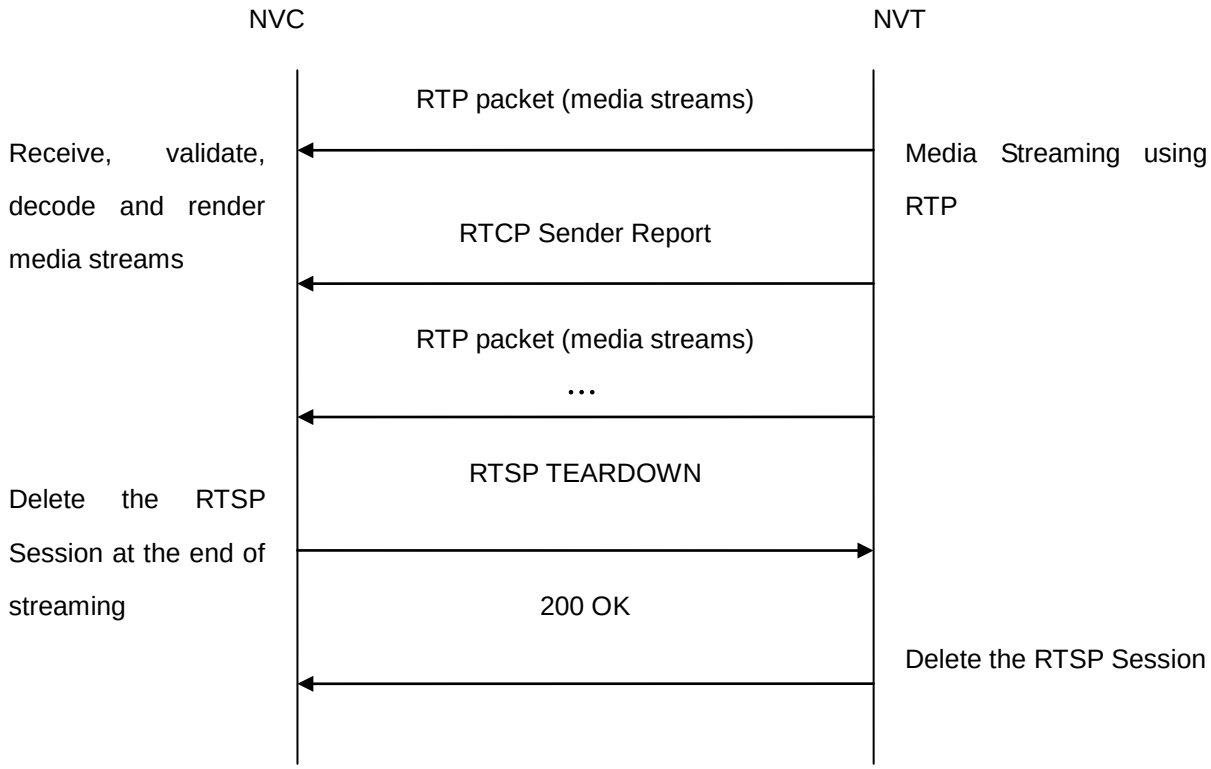
Pre-Requisite: MPEG4 is implemented by NVT

A media profile with MPEG4 video encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.14
Select media profile
SetVideoEncoderConfigurationRequest
(MPEG4 Video Encoder Cfg, FALSE)
Set MPEG4 video encoder cfg
Modify MPEG4 video encoder cfg
SetVideoEncoderConfigurationResponse
GetStreamUriRequest
(Profile Token, RTP-Unicast, UDP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Receive and Validate SDP message
200 OK (SDP Message)
Send SDP message
RTSP SETUP
Send Stream Information
Receive and Validate Stream information
200 OK (Media Stream Information)
RTSP PLAY
Initiate Media Streaming
Ready for Media Streaming
200 OK (RTP-Info)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with MPEG4 video encoding support by following the procedure mentioned in Annex A.14.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "MPEG4", Resolution = ["Width", "Height"], Quality = q1, GovLength = g1, Mpeg4Profile = SP, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.14.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, UDP transport**) to retrieve media stream URI for the selected media profile.
7. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
8. NVC verifies the RTSP media stream URI provided by the NVT.
9. NVC invokes RTSP DESCRIBE request.
10. NVT sends 200 OK message and SDP information.

11. NVC invokes RTSP SETUP request with transport parameter as **RTP/UDP**.
12. NVT sends 200 OK message and the media stream information.
13. NVC invokes RTSP PLAY request.
14. NVT sends 200 OK message and starts media streaming.
15. NVT sends MPEG4 RTP media stream to NVC over UDP.
16. NVT sends RTCP sender report to NVC.
17. NVT validates the received RTP and RTCP packets, decodes and renders them.
18. NVC invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.
19. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.5 for Invalid RTP header definition.

8.1.7 NVT MEDIA STREAMING – MPEG4 (RTP-Unicast/RTSP/HTTP/TCP)

Test Label: Real Time Viewing NVT MPEG4 media streaming using HTTP transport.

ONVIF Core Specification Coverage: 11.1.1.4 RTP/RTSP/HTTP/TCP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP, 11.2.1.2 RTSP over HTTP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF IMPLEMENTED (MPEG4-SP)

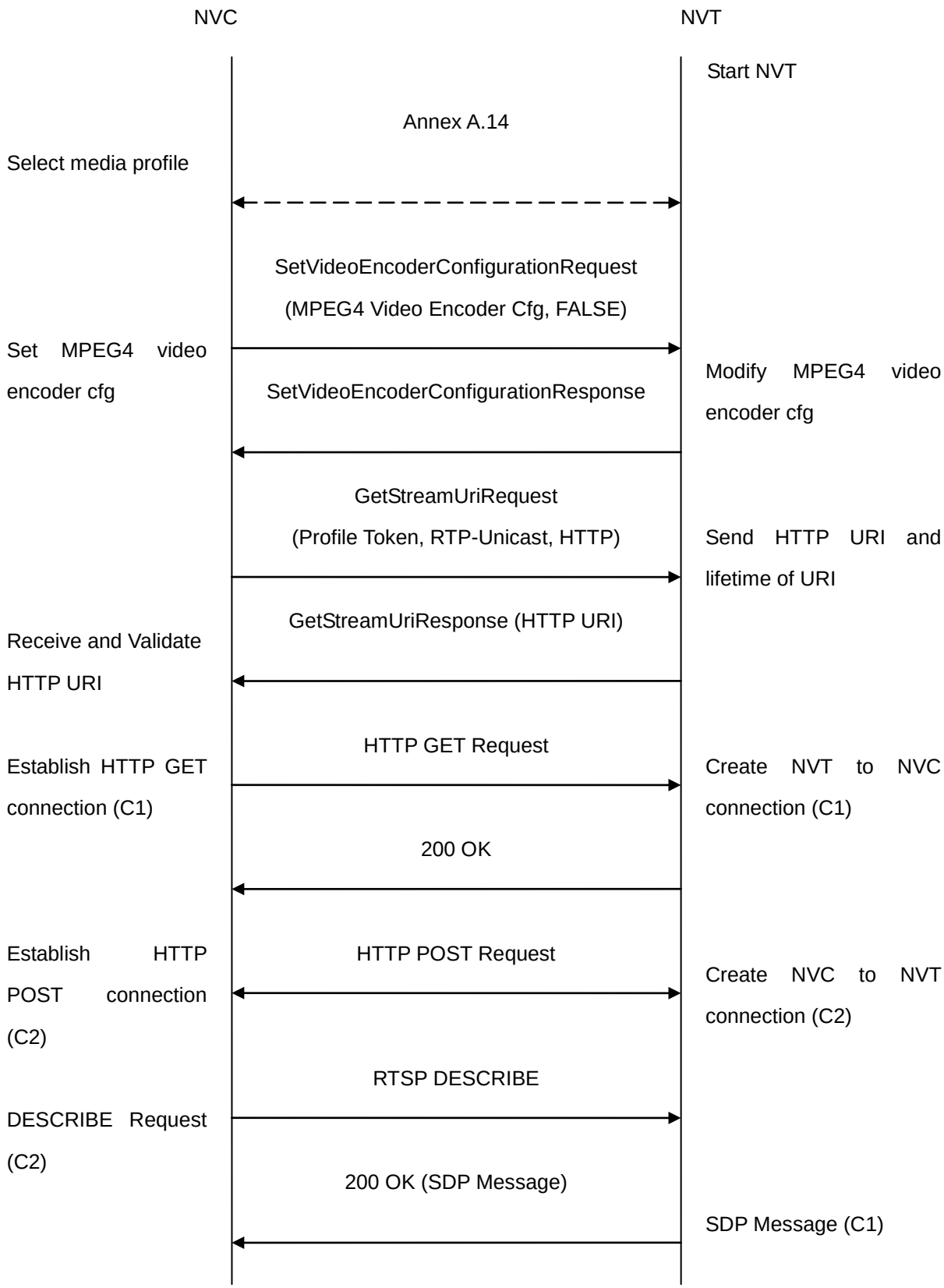
Test Purpose: To verify MPEG4 media streaming based on HTTP Transport.

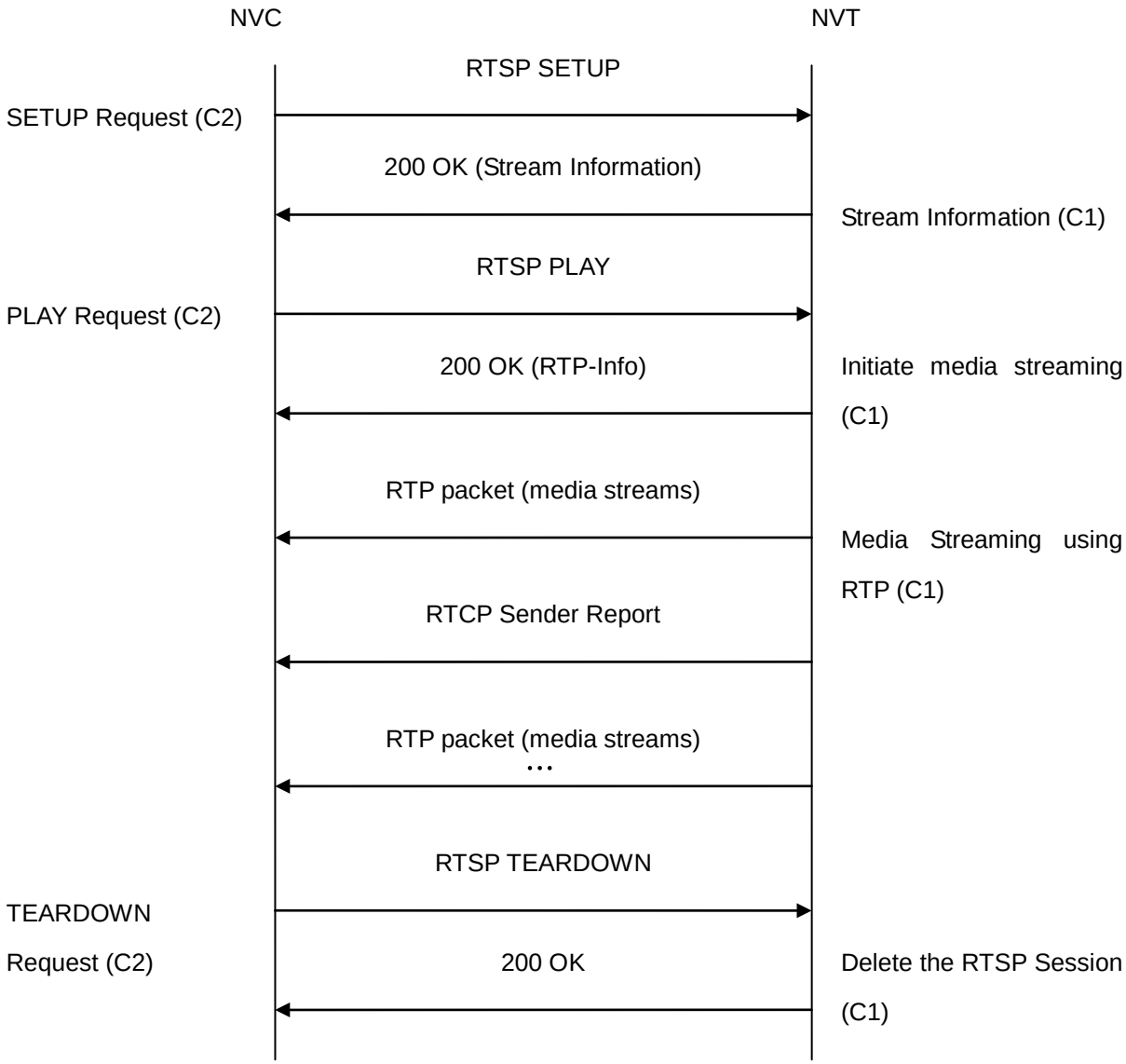
Pre-Requisite: MPEG4 is implemented by NVT.

A media profile with MPEG4 video encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.14
Select media profile
SetVideoEncoderConfigurationRequest
(MPEG4 Video Encoder Cfg, FALSE)
Set MPEG4 video encoder cfg
Modify MPEG4 video encoder cfg
SetVideoEncoderConfigurationResponse
GetStreamUriRequest
(Profile Token, RTP-Unicast, HTTP)
Send HTTP URI and lifetime of URI
GetStreamUriResponse (HTTP URI)
Receive and Validate HTTP URI
HTTP GET Request
Establish HTTP GET connection (C1)
Create NVT to NVC connection (C1)
200 OK
Establish HTTP POST connection (C2)
HTTP POST Request
Create NVC to NVT connection (C2)
RTSP DESCRIBE
DESCRIBE Request (C2)
200 OK (SDP Message)
SDP Message (C1)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with MPEG4 video encoding support by following the procedure mentioned in Annex A.14.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "MPEG4", Resolution = ["Width", "Height"], Quality = q1, GovLength = g1, Mpeg4Profile = SP, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.14.

5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, HTTP transport**) to retrieve media stream URI for the selected media profile.
7. NVT sends HTTP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
8. NVC verifies the HTTP media stream URI provided by the NVT.
9. NVC invokes HTTP GET Request on NVT and establishes NVT to NVC connection for RTP data transfer.
10. NVC invokes HTTP POST Request and establishes NVC to NVT connection for RTSP control requests.
11. NVC invokes RTSP DESCRIBE request on HTTP POST connection.
12. NVT sends 200 OK message and SDP information on HTTP GET connection.
13. NVC invokes RTSP SETUP request on HTTP POST connection with transport parameter as **'RTP/TCP'** along with **'interleaved'** parameter.
14. NVT sends 200 OK message and the media stream information on HTTP GET connection.
15. NVC invokes RTSP PLAY request on HTTP POST connection.
16. NVT sends 200 OK message and starts media streaming on HTTP GET connection.
17. NVT transfers MPEG4 RTP media stream to NVC on HTTP GET connection.
18. NVT sends RTCP sender report to NVC on HTTP GET connection.
19. NVT validates the received RTP and RTCP packets, decodes and renders them.
20. NVC invokes RTSP TEARDOWN control request on HTTP POST connection and closes the HTTP POST connection.
21. NVT sends 200 OK Response on HTTP GET connection and closes the HTTP GET connection.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – HTTP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.5 for Invalid RTP header definition.

8.1.8 NVT MEDIA STREAMING – MPEG4 (RTP/RTSP/TCP)

Test Label: Real Time Viewing NVT MPEG4 media streaming using RTP/RTSP/TCP transport.

ONVIF Core Specification Coverage: 11.1.1.3 RTP/RTSP/TCP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF IMPLEMENTED (MPEG4-SP & RTP/RTSP/TCP)

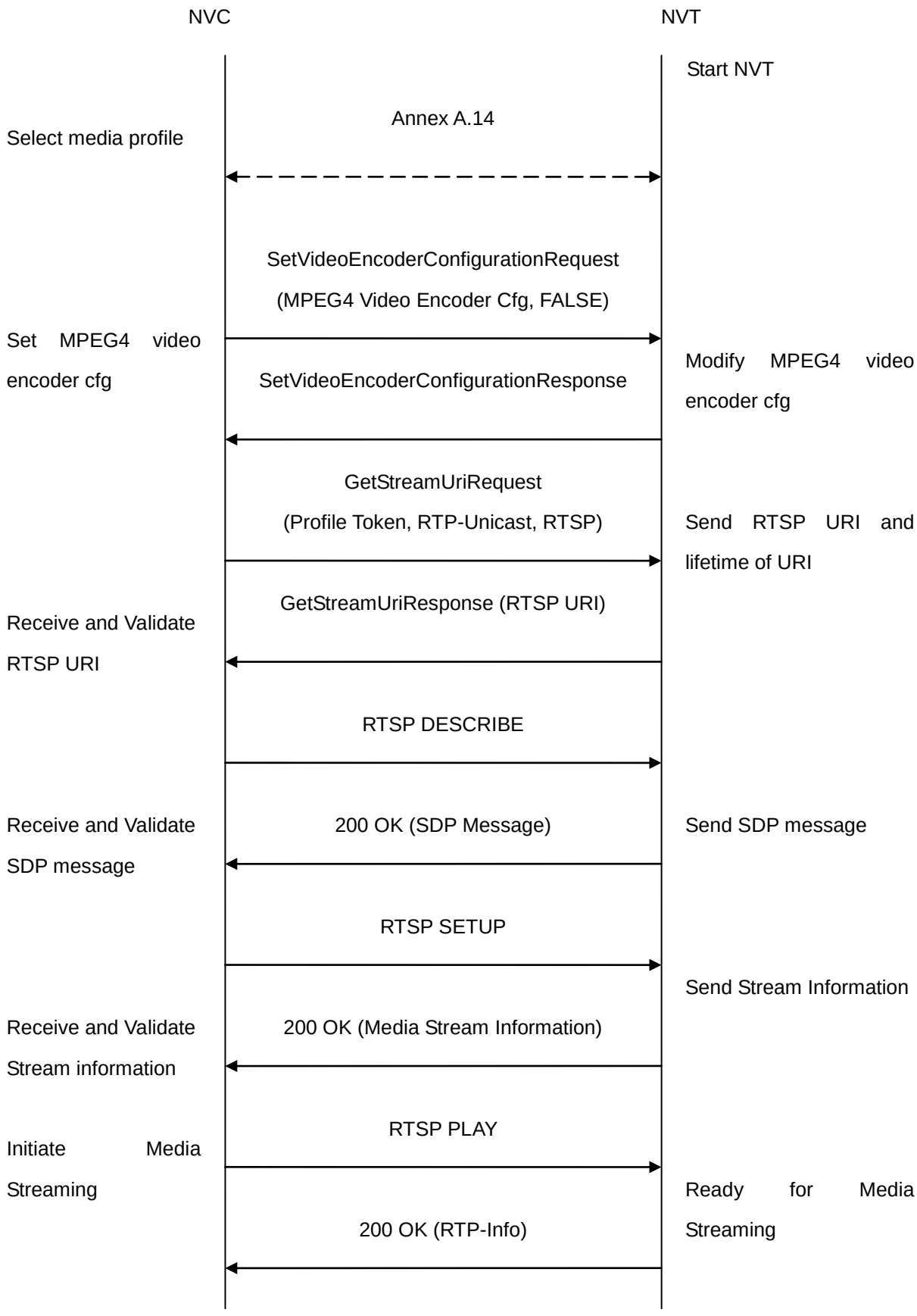
Test Purpose: To verify MPEG4 media streaming based on RTP/RTSP/TCP using RTSP tunnel.

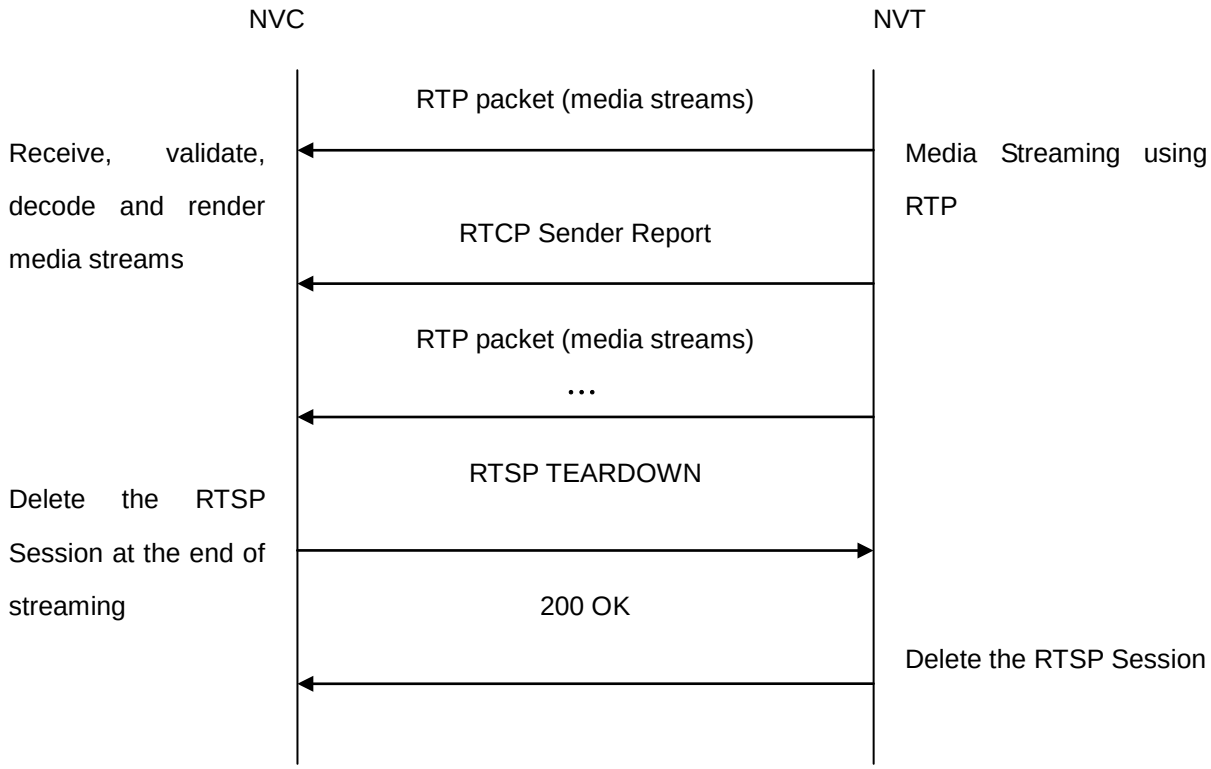
Pre-Requisite: MPEG4 and RTP/RTSP/TCP media streaming is implemented by NVT.

A media profile with MPEG4 video encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.14
Select media profile
SetVideoEncoderConfigurationRequest
(MPEG4 Video Encoder Cfg, FALSE)
Set MPEG4 video encoder cfg
Modify MPEG4 video encoder cfg
SetVideoEncoderConfigurationResponse
GetStreamUriRequest
(Profile Token, RTP-Unicast, RTSP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Receive and Validate SDP message
200 OK (SDP Message)
Send SDP message
RTSP SETUP
Send Stream Information
Receive and Validate Stream information
200 OK (Media Stream Information)
RTSP PLAY
Initiate Media Streaming
Ready for Media Streaming
200 OK (RTP-Info)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with MPEG4 video encoding support by following the procedure mentioned in Annex A.14.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "MPEG4", Resolution = ["Width", "Height"], Quality = q1, GovLength = g1, Mpeg4Profile = SP, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.14.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, RTSP transport**) to retrieve media stream URI for the selected media profile.
7. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
8. NVC verifies the RTSP media stream URI provided by the NVT.
9. NVC invokes RTSP DESCRIBE request.
10. NVT sends 200 OK message and SDP information.

11. NVC invokes RTSP SETUP request with transport parameter as **'RTP/TCP'** along with **'interleaved'** parameter.
12. NVT sends 200 OK message and the media stream information.
13. NVC invokes RTSP PLAY request.
14. NVT sends 200 OK message and starts media streaming.
15. NVT interleaves RTP and RTCP packets, send them over RTSP control connection.
16. NVT validates the received RTP and RTCP packets, decodes and renders them.
17. NVC invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.
18. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send GetProfilesResponse message.

DUT did not have valid media profile.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send RTP and RTCP packets as per [RFC 2326] section 10.12.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

8.1.9 NVT SET SYNCHRONIZATION POINT – MPEG4

Test Label: Media Configuration NVT Synchronization Point – MPEG4

ONVIF Core Specification Coverage: 10.14.1 Set synchronization point.

Device Type: NVT

Command Under Test: SetSynchronizationPoint

WSDL Reference: media.wsdl

Requirement Level: MUST IF IMPLEMENTED (MPEG4-SP)

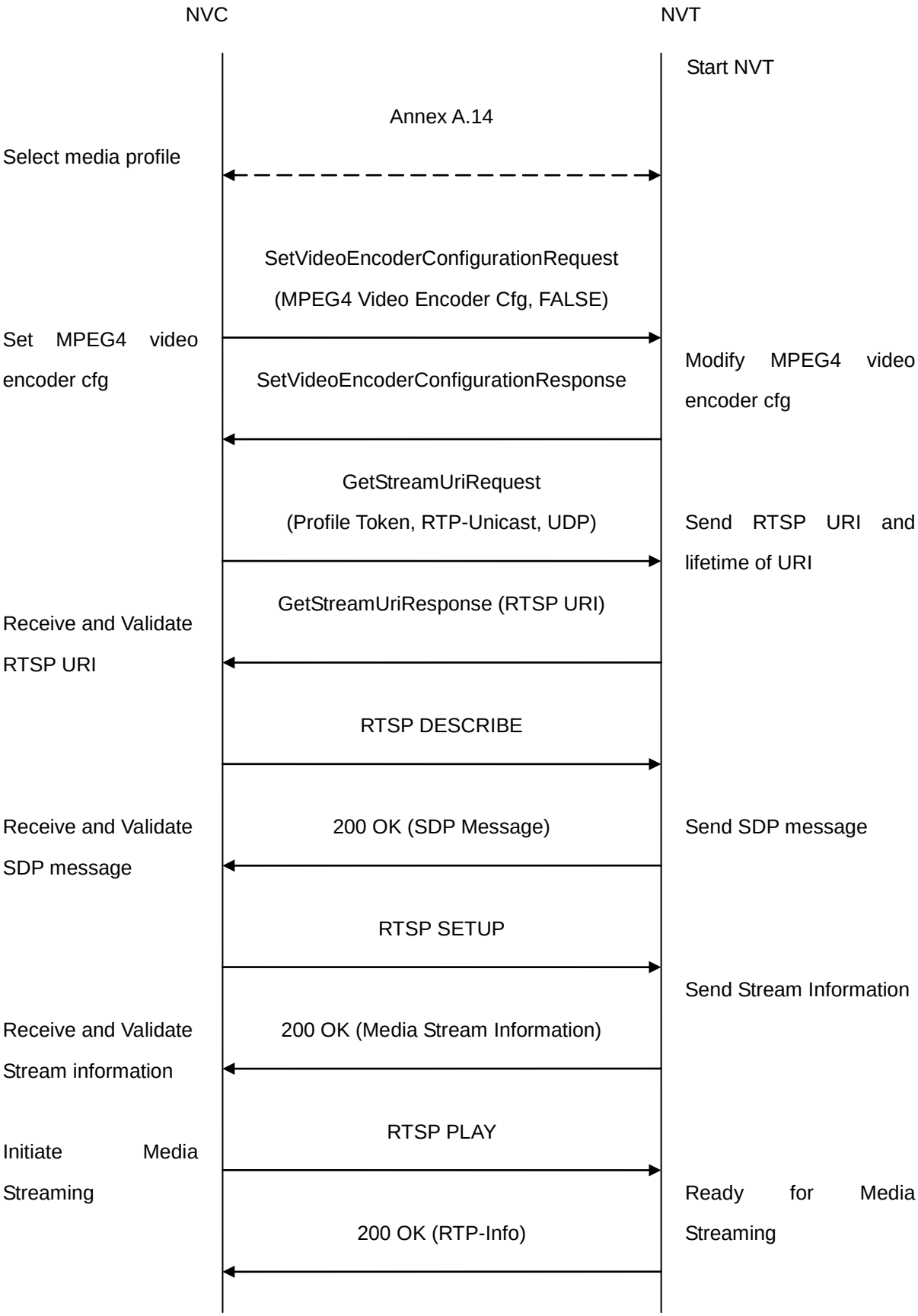
Test Purpose: To request synchronization point from NVT for MPEG4 media stream.

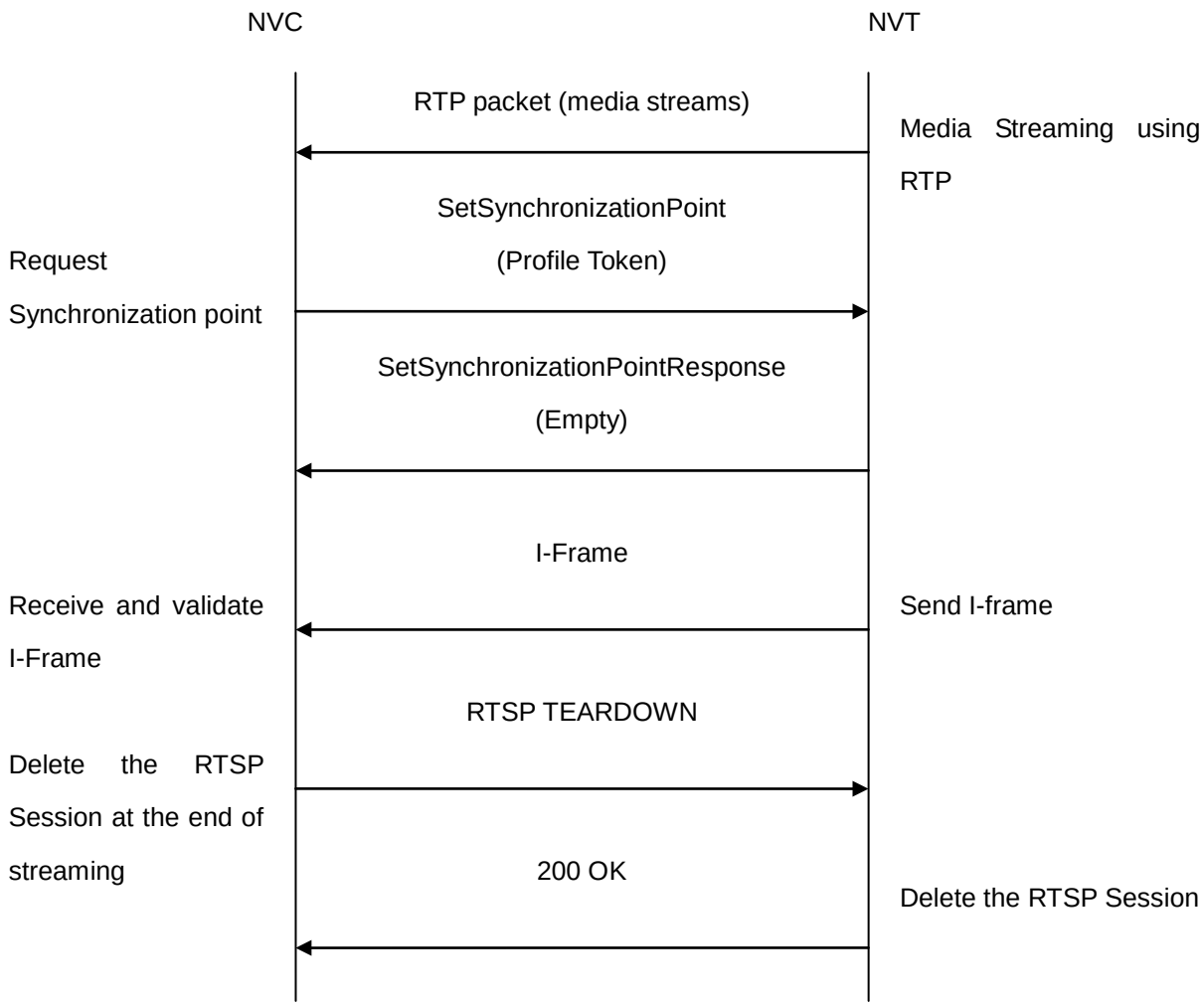
Pre-Requisite: MPEG4 is implemented by NVT.

A media profile with MPEG4 video encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.14
Select media profile
SetVideoEncoderConfigurationRequest
(MPEG4 Video Encoder Cfg, FALSE)
Set MPEG4 video encoder cfg
Modify MPEG4 video encoder cfg
SetVideoEncoderConfigurationResponse
GetStreamUriRequest
(Profile Token, RTP-Unicast, UDP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Receive and Validate SDP message
200 OK (SDP Message)
Send SDP message
RTSP SETUP
Send Stream Information
Receive and Validate Stream information
200 OK (Media Stream Information)
RTSP PLAY
Initiate Media Streaming
Ready for Media Streaming
200 OK (RTP-Info)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with MPEG4 video encoding support by following the procedure mentioned in Annex A.14.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "MPEG4", Resolution = ["Width", "Height"], Quality = q1, GovLength = g1, Mpeg4Profile = SP, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.14.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, UDP transport**) to retrieve media stream URI for the selected media profile.

7. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
8. NVC verifies the RTSP media stream URI provided by the NVT.
9. NVC invokes RTSP DESCRIBE request.
10. NVT sends 200 OK message and SDP information.
11. NVC invokes RTSP SETUP request with transport parameter as **RTP/UDP**.
12. NVT sends 200 OK message and the media stream information.
13. NVC invokes RTSP PLAY request.
14. NVT sends 200 OK message and starts media streaming.
15. NVT sends MPEG4 RTP media stream to NVC over UDP.
16. NVT sends RTCP sender report to NVC.
17. NVT validates the received RTP and RTCP packets, decodes and renders them.
18. NVC invokes SetSynchronizationPoint request on the selected media profile.
19. NVT sends the SetSynchronizationPoint response indicating success.
20. NVT inserts the I-frame in the on going media stream.
21. NVC verifies that I-frame is sent by NVT before the regular 'I-frame insertion time interval'.
22. NVC invokes RTSP TEARDOWN control request to terminate the RTSP session.
23. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send SetSynchronizationPointResponse message.

DUT did not send I-frame before the regular 'I-frame insertion time interval' upon invoking SetSynchronizationPoint request.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.11 for details on 'I-frame insertion time interval'.

8.1.10 NVT MEDIA STREAMING – H.264 (RTP-Unicast/ UDP)

Test Label: Real Time Viewing NVT H.264 media streaming using RTP-Unicast/UDP transport.

ONVIF Core Specification Coverage: 11.1.1.1 RTP data transfer via UDP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF IMPLEMENTED (H.264-Baseline)

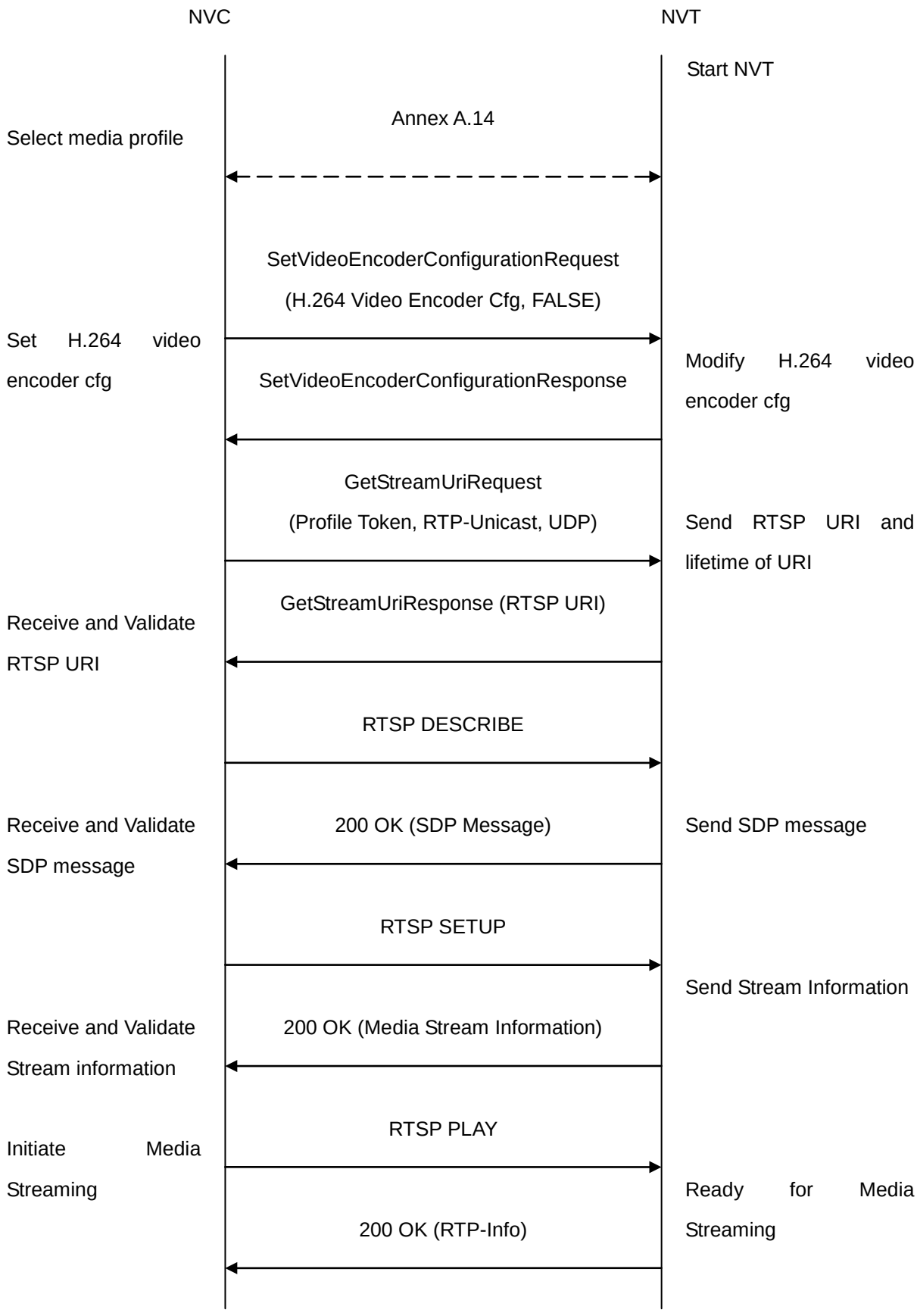
Test Purpose: To verify H.264 media streaming based on RTP/UDP Unicast Transport.

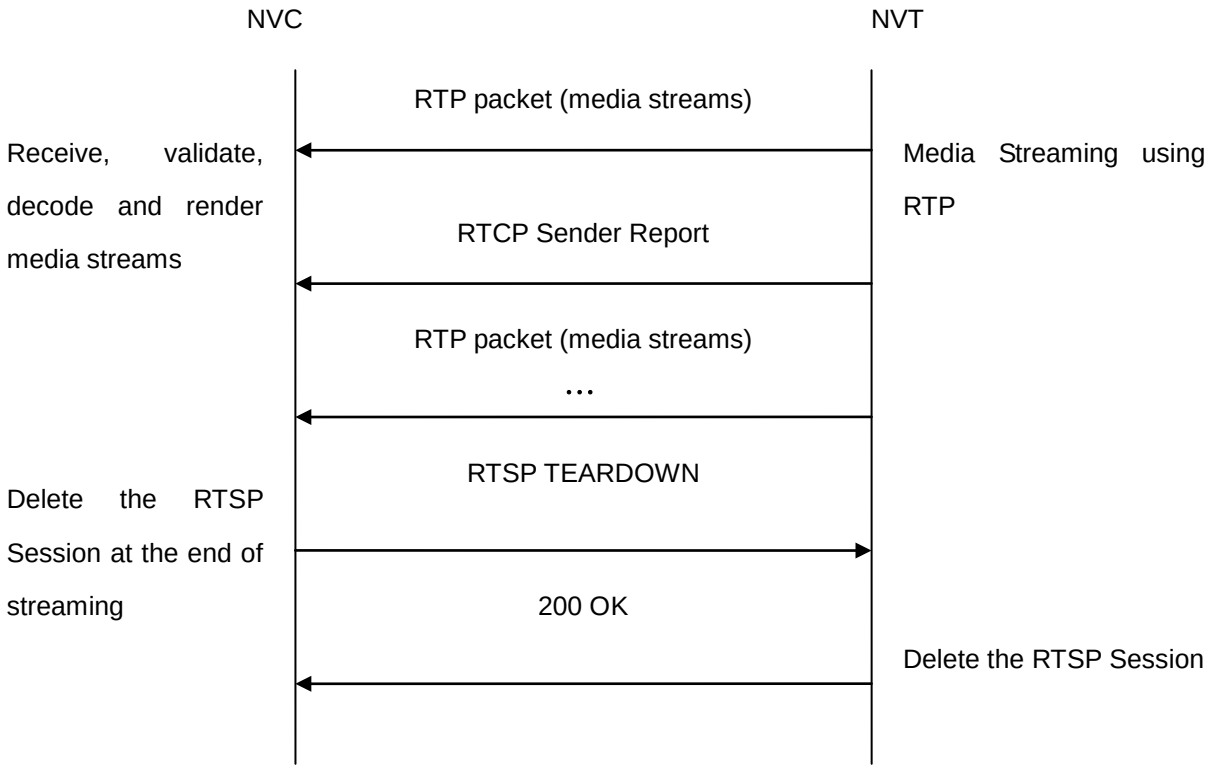
Pre-Requisite: H.264 is implemented by NVT

A media profile with H.264 video encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.14
Select media profile
SetVideoEncoderConfigurationRequest
(H.264 Video Encoder Cfg, FALSE)
Set H.264 video encoder cfg
Modify H.264 video encoder cfg
SetVideoEncoderConfigurationResponse
GetStreamUriRequest
(Profile Token, RTP-Unicast, UDP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Receive and Validate SDP message
200 OK (SDP Message)
Send SDP message
RTSP SETUP
Send Stream Information
Receive and Validate Stream information
200 OK (Media Stream Information)
RTSP PLAY
Initiate Media Streaming
Ready for Media Streaming
200 OK (RTP-Info)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with H.264 video encoding support by following the procedure mentioned in Annex A.14.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "H.264", Resolution = ["Width", "Height"], Quality = q1, GovLength = g1, H264Profile = Baseline, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.14.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, UDP transport**) to retrieve media stream URI for the selected media profile.
7. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
8. NVC verifies the RTSP media stream URI provided by the NVT.
9. NVC invokes RTSP DESCRIBE request.
10. NVT sends 200 OK message and SDP information.

11. NVC invokes RTSP SETUP request with transport parameter as **RTP/UDP**.
12. NVT sends 200 OK message and the media stream information.
13. NVC invokes RTSP PLAY request.
14. NVT sends 200 OK message and starts media streaming.
15. NVT sends H.264 RTP media stream to NVC over UDP.
16. NVT sends RTCP sender report to NVC.
17. NVT validates the received RTP and RTCP packets, decodes and renders them.
18. NVC invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.
19. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.5 for Invalid RTP header definition.

8.1.11 NVT MEDIA STREAMING – H.264 (RTP-Unicast/RTSP/HTTP/TCP)

Test Label: Real Time Viewing NVT H.264 media streaming using HTTP transport.

ONVIF Core Specification Coverage: 11.1.1.4 RTP/RTSP/HTTP/TCP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP, 11.2.1.2 RTSP over HTTP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF IMPLEMENTED (H.264-Baseline)

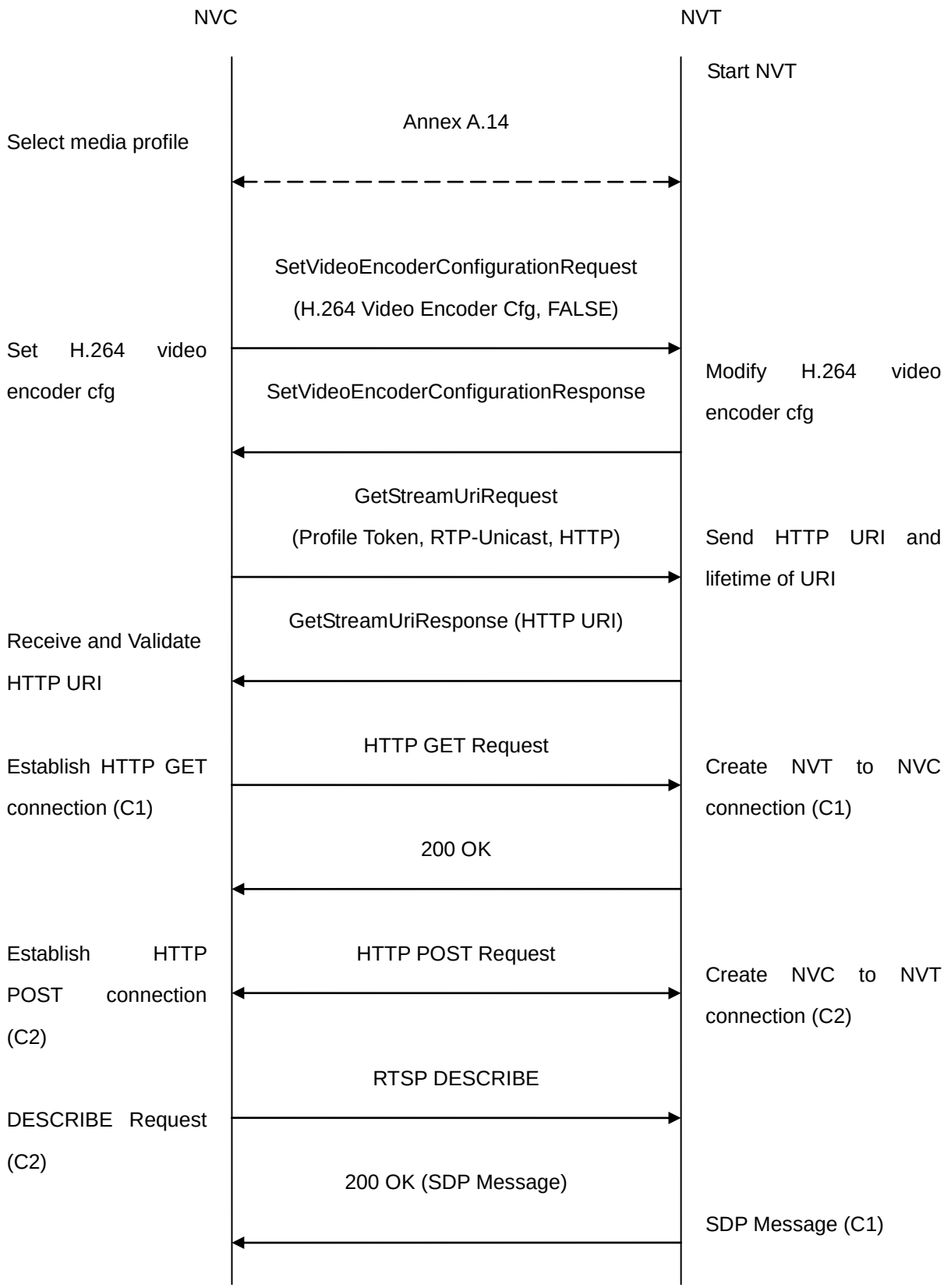
Test Purpose: To verify H.264 media streaming based on HTTP Transport.

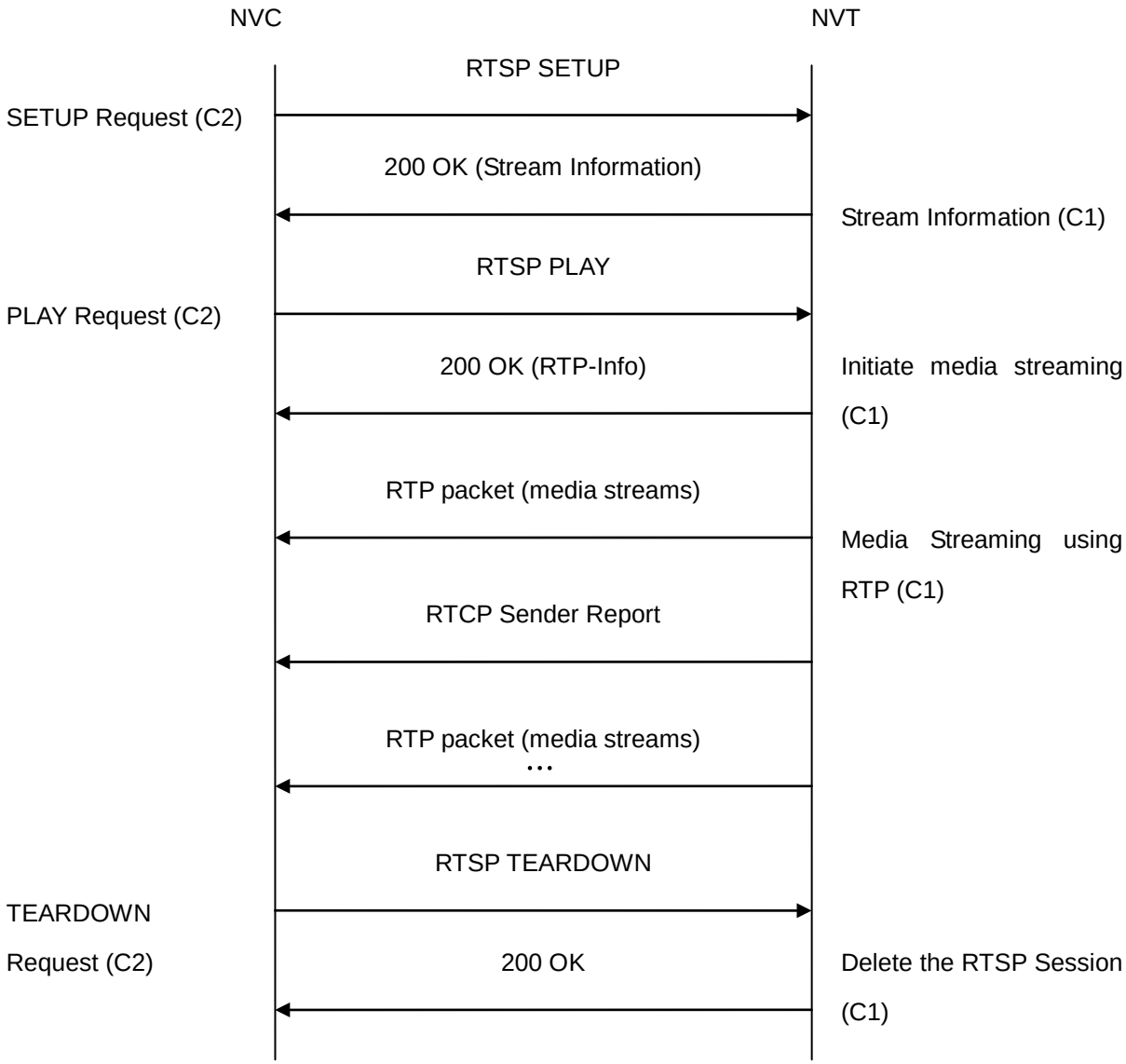
Pre-Requisite: H.264 is implemented by NVT.

A media profile with H.264 video encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.14
Select media profile
SetVideoEncoderConfigurationRequest
(H.264 Video Encoder Cfg, FALSE)
Set H.264 video encoder cfg
Modify H.264 video encoder cfg
SetVideoEncoderConfigurationResponse
GetStreamUriRequest
(Profile Token, RTP-Unicast, HTTP)
Send HTTP URI and lifetime of URI
GetStreamUriResponse (HTTP URI)
Receive and Validate HTTP URI
HTTP GET Request
Establish HTTP GET connection (C1)
Create NVT to NVC connection (C1)
200 OK
Establish HTTP POST connection (C2)
HTTP POST Request
Create NVC to NVT connection (C2)
RTSP DESCRIBE
DESCRIBE Request (C2)
200 OK (SDP Message)
SDP Message (C1)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with H.264 video encoding support by following the procedure mentioned in Annex A.14.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "H.264", Resolution = ["Width", "Height"], Quality = q1, GovLength = g1, H264Profile = Baseline, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.14.

5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, HTTP transport**) to retrieve media stream URI for the selected media profile.
7. NVT sends HTTP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
8. NVC verifies the HTTP media stream URI provided by the NVT.
9. NVC invokes HTTP GET Request on NVT and establishes NVT to NVC connection for RTP data transfer.
10. NVC invokes HTTP POST Request and establishes NVC to NVT connection for RTSP control requests.
11. NVC invokes RTSP DESCRIBE request on HTTP POST connection.
12. NVT sends 200 OK message and SDP information on HTTP GET connection.
13. NVC invokes RTSP SETUP request on HTTP POST connection with transport parameter as **'RTP/TCP'** along with **'interleaved'** parameter.
14. NVT sends 200 OK message and the media stream information on HTTP GET connection.
15. NVC invokes RTSP PLAY request on HTTP POST connection.
16. NVT sends 200 OK message and starts media streaming on HTTP GET connection.
17. NVT transfers H.264 RTP media stream to NVC on HTTP GET connection.
18. NVT sends RTCP sender report to NVC on HTTP GET connection.
19. NVT validates the received RTP and RTCP packets, decodes and renders them.
20. NVC invokes RTSP TEARDOWN control request on HTTP POST connection and closes the HTTP POST connection.
21. NVT sends 200 OK Response on HTTP GET connection and closes the HTTP GET connection.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – HTTP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.5 for Invalid RTP header definition.

8.1.12 NVT MEDIA STREAMING – H.264 (RTP/RTSP/TCP)

Test Label: Real Time Viewing NVT H.264 media streaming using RTP/RTSP/TCP transport.

ONVIF Core Specification Coverage: 11.1.1.3 RTP/RTSP/TCP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF IMPLEMENTED (H.264-Baseline & RTP/RTSP/TCP)

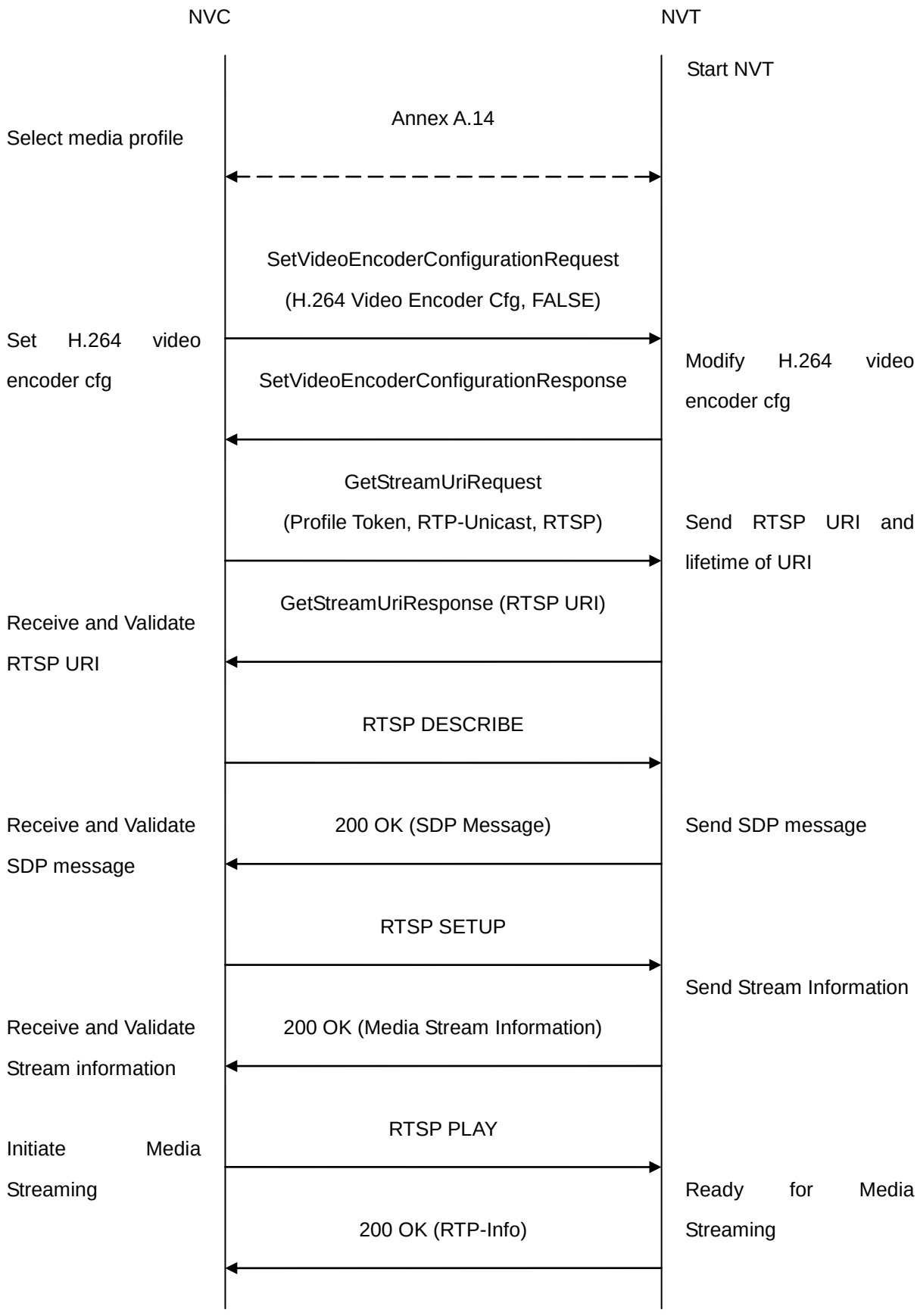
Test Purpose: To verify H.264 media streaming based on RTP/RTSP/TCP using RTSP tunnel.

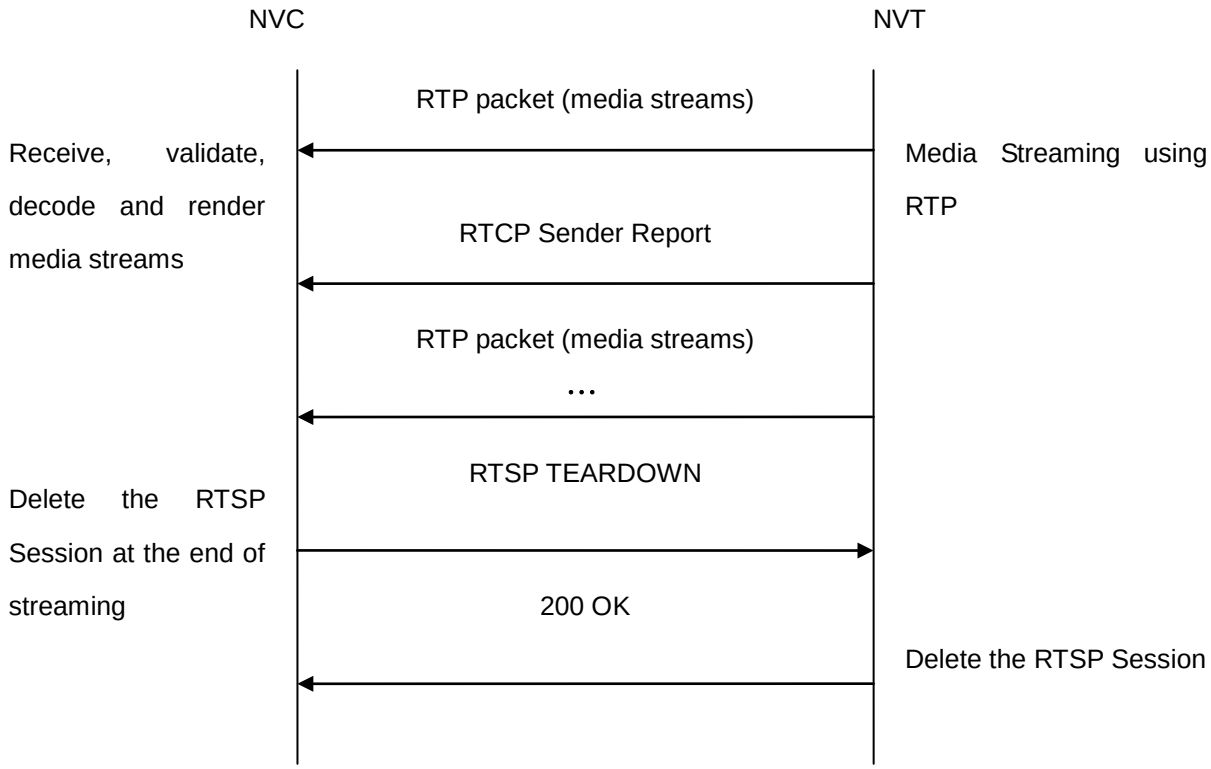
Pre-Requisite: H.264 and RTP/RTSP/TCP media streaming is implemented by NVT.

A media profile with H.264 video encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.14
Select media profile
SetVideoEncoderConfigurationRequest
(H.264 Video Encoder Cfg, FALSE)
Set H.264 video encoder cfg
Modify H.264 video encoder cfg
SetVideoEncoderConfigurationResponse
GetStreamUriRequest
(Profile Token, RTP-Unicast, RTSP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Receive and Validate SDP message
200 OK (SDP Message)
Send SDP message
RTSP SETUP
Send Stream Information
Receive and Validate Stream information
200 OK (Media Stream Information)
RTSP PLAY
Initiate Media Streaming
Ready for Media Streaming
200 OK (RTP-Info)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with H.264 video encoding support by following the procedure mentioned in Annex A.14.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "H.264", Resolution = ["Width", "Height"], Quality = q1, GovLength = g1, H264Profile = Baseline, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.14.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, RTSP transport**) to retrieve media stream URI for the selected media profile.
7. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
8. NVC verifies the RTSP media stream URI provided by the NVT.
9. NVC invokes RTSP DESCRIBE request.
10. NVT sends 200 OK message and SDP information.

11. NVC invokes RTSP SETUP request with transport parameter as **'RTP/TCP'** along with **'interleaved'** parameter.
12. NVT sends 200 OK message and the media stream information.
13. NVC invokes RTSP PLAY request.
14. NVT sends 200 OK message and starts media streaming.
15. NVT interleaves RTP and RTCP packets, send them over RTSP control connection.
16. NVT validates the received RTP and RTCP packets, decodes and renders them.
17. NVC invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.
18. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send RTP and RTCP packets as per [RFC 2326] section 10.12.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

8.1.13 NVT SET SYNCHRONIZATION POINT – H.264

Test Label: Media Configuration NVT Synchronization Point – H.264

ONVIF Core Specification Coverage: 10.14.1 Set synchronization point.

Device Type: NVT

Command Under Test: SetSynchronizationPoint

WSDL Reference: media.wsdl

Requirement Level: MUST IF IMPLEMENTED (H.264-Baseline)

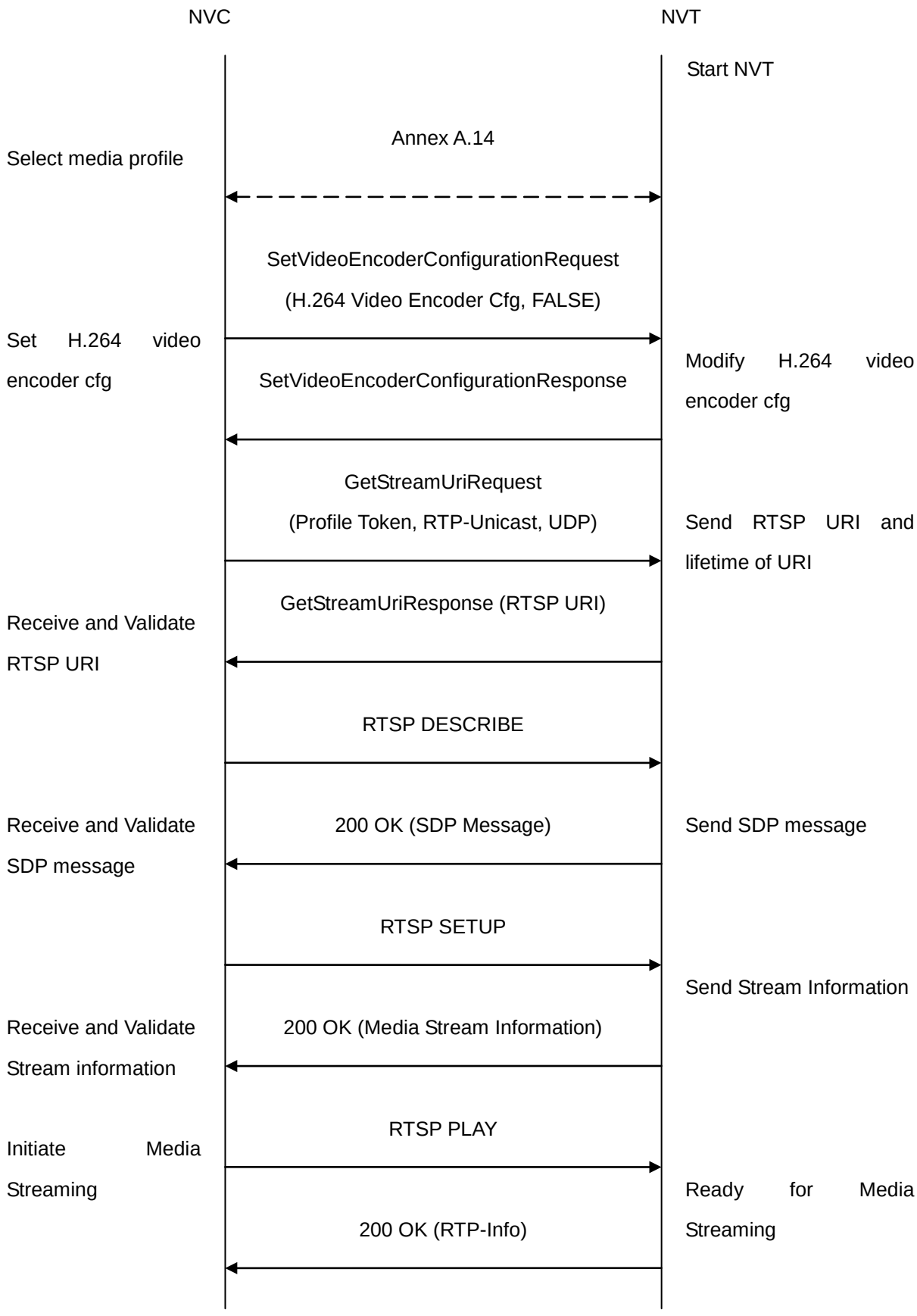
Test Purpose: To request synchronization point from NVT for H.264 media stream.

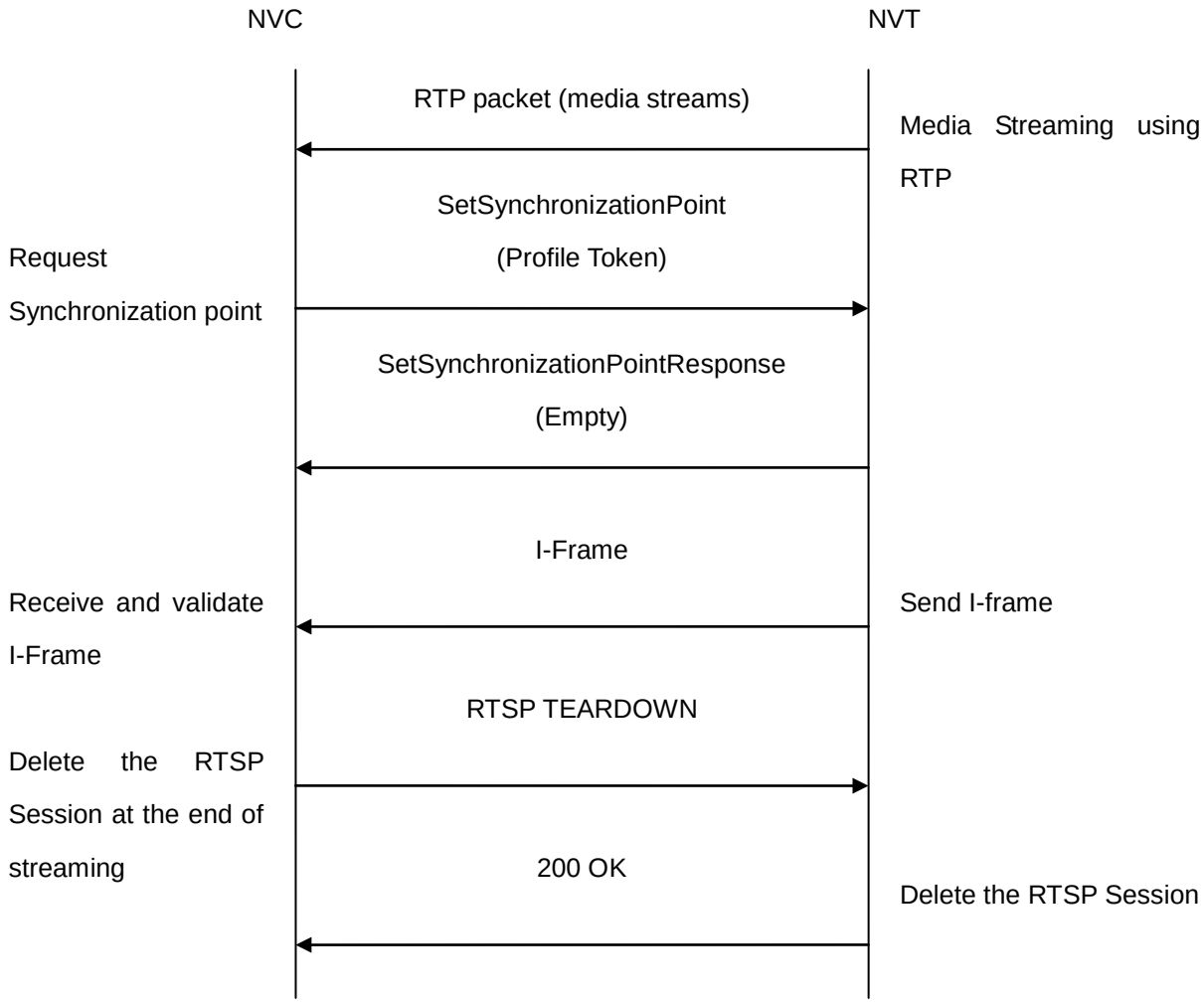
Pre-Requisite: H.264 is implemented by NVT.

A media profile with H.264 video encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.14
Select media profile
SetVideoEncoderConfigurationRequest
(H.264 Video Encoder Cfg, FALSE)
Set H.264 video encoder cfg
SetVideoEncoderConfigurationResponse
Modify H.264 video encoder cfg
GetStreamUriRequest
(Profile Token, RTP-Unicast, UDP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Receive and Validate SDP message
200 OK (SDP Message)
Send SDP message
RTSP SETUP
Send Stream Information
Receive and Validate Stream information
200 OK (Media Stream Information)
RTSP PLAY
Initiate Media Streaming
Ready for Media Streaming
200 OK (RTP-Info)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with H.264 video encoding support by following the procedure mentioned in Annex A.14.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "H.264", Resolution = ["Width", "Height"], Quality = q1, GovLength = g1, H264Profile = Baseline, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.14.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, UDP transport**) to retrieve media stream URI for the selected media profile.

7. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
8. NVC verifies the RTSP media stream URI provided by the NVT.
9. NVC invokes RTSP DESCRIBE request.
10. NVT sends 200 OK message and SDP information.
11. NVC invokes RTSP SETUP request with transport parameter as **RTP/UDP**.
12. NVT sends 200 OK message and the media stream information.
13. NVC invokes RTSP PLAY request.
14. NVT sends 200 OK message and starts media streaming.
15. NVT sends H.264 RTP media stream to NVC over UDP.
16. NVT sends RTCP sender report to NVC.
17. NVT validates the received RTP and RTCP packets, decodes and renders them.
18. NVC invokes SetSynchronizationPoint request on the selected media profile.
19. NVT sends the SetSynchronizationPointResponse indicating success.
20. NVT inserts the I-frame in the on going media stream.
21. NVC verifies that I-frame is sent by NVT before the regular 'I-frame insertion time interval'.
22. NVC invokes RTSP TEARDOWN control request to terminate the RTSP session.
23. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send SetSynchronizationPointResponse message.

DUT did not send I-frame before the regular 'I-frame insertion time interval' upon invoking SetSynchronizationPoint request.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.11 for details on 'I-frame insertion time interval'.

8.2 Audio & Video Streaming

8.2.1 NVT MEDIA STREAMING – JPEG/G.711 (RTP-Unicast/ UDP)

Test Label: Real Time Viewing NVT JPEG/G.711 Audio&Video streaming using RTP-Unicast/UDP transport.

ONVIF Core Specification Coverage: 11.1.1.1 RTP data transfer via UDP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF SUPPORTED (Audio)

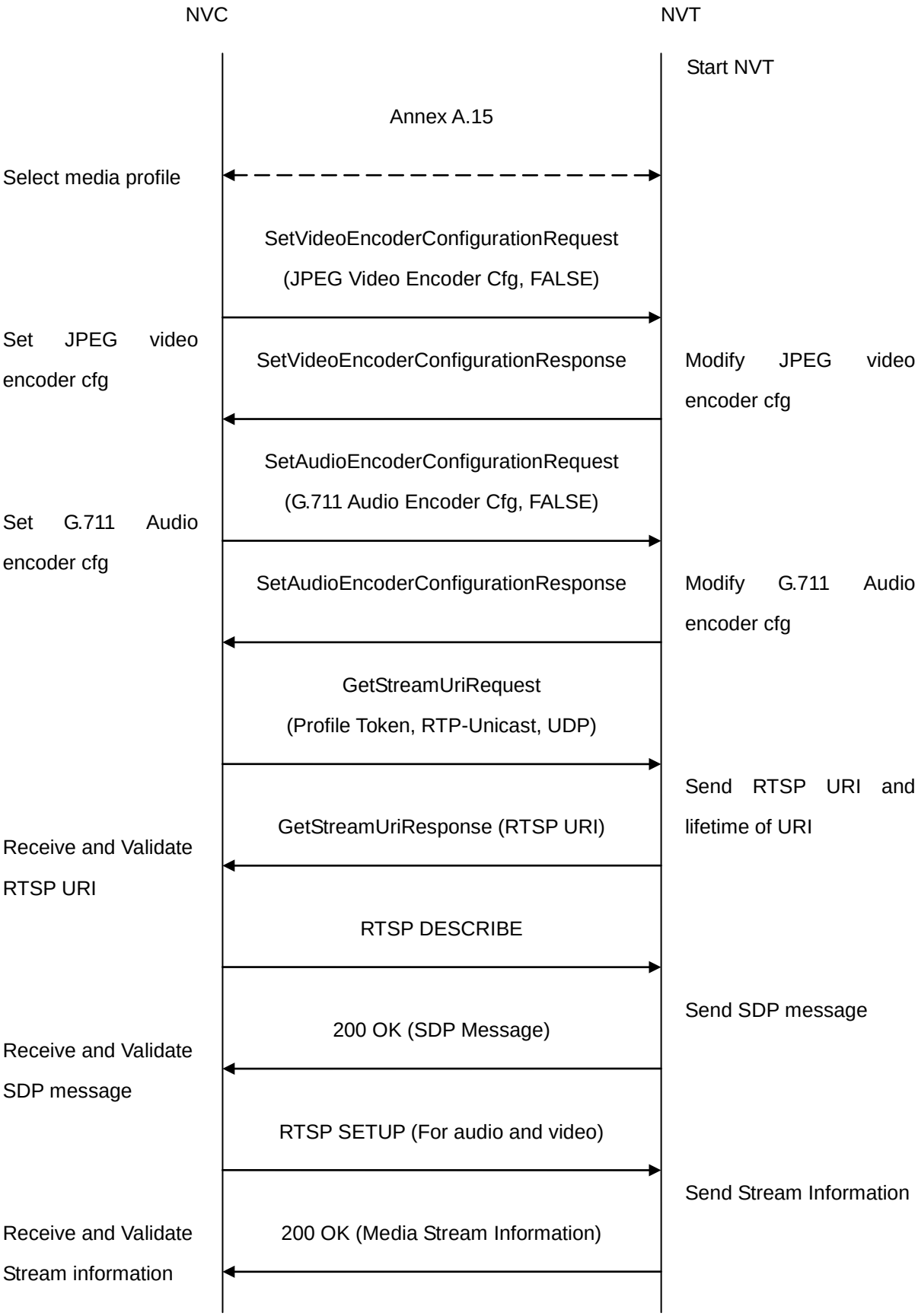
Test Purpose: To verify JPEG/G.711 Audio&Video streaming based on RTP/UDP Unicast Transport.

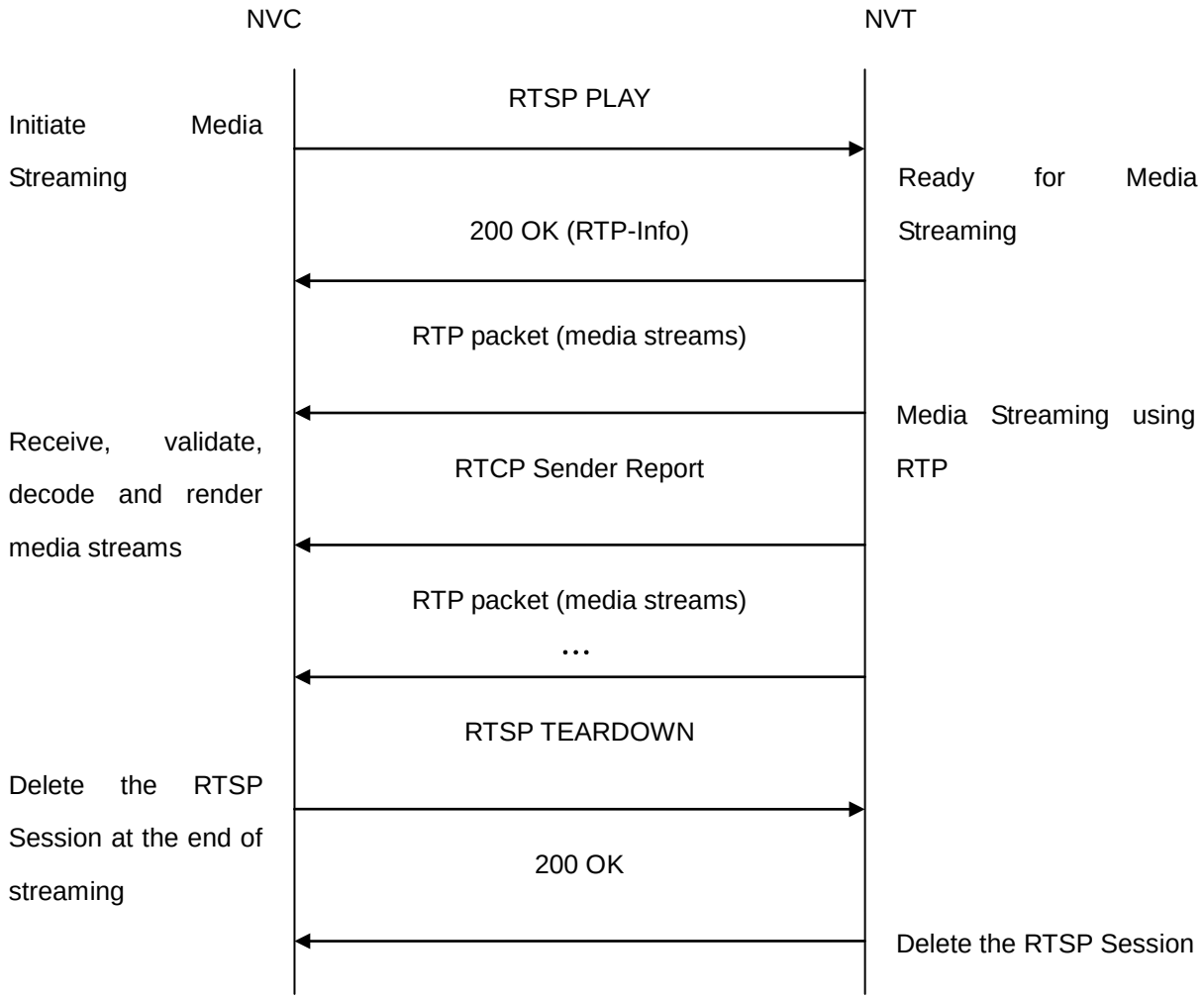
Pre-Requisite: Audio is supported by NVT

A media profile with JPEG video encoder configuration and G.711 audio encoder configuration

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.15
Select media profile
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
Modify JPEG video encoder cfg
SetAudioEncoderConfigurationRequest
(G.711 Audio Encoder Cfg, FALSE)
Set G.711 Audio encoder cfg
SetAudioEncoderConfigurationResponse
Modify G.711 Audio encoder cfg
GetStreamUriRequest
(Profile Token, RTP-Unicast, UDP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Send SDP message
200 OK (SDP Message)
Receive and Validate SDP message
RTSP SETUP (For audio and video)
Send Stream Information
200 OK (Media Stream Information)
Receive and Validate Stream information



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with both JPEG video encoding support and G.711 audio encoding support by following the procedure mentioned in Annex A.15.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.15.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes SetAudioEncoderConfigurationRequest (**Encoding = "G711", Bitrate = r1, SampleRate = r2, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetAudioEncoderConfigurationOptions response in A.15.
7. NVT modifies audio encoder configuration and responds with SetAudioEncoderConfigurationResponse message indicating success.

8. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, UDP transport**) to retrieve media stream URI for the selected media profile.
9. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
10. NVC verifies the RTSP media stream URI provided by the NVT.
11. NVC invokes RTSP DESCRIBE request.
12. NVT sends 200 OK message and SDP information.
13. NVC invokes RTSP SETUP request with transport parameter as **RTP/UDP** for both audio and video streams separately.
14. NVT sends 200 OK message and the media stream information.
15. NVC invokes RTSP PLAY request.
16. NVT sends 200 OK message and starts media streaming.
17. NVT sends Audio/Video RTP media stream to NVC over UDP.
18. NVT sends Audio/Video RTCP sender report to NVC.
19. NVT validates the received RTP and RTCP packets, decodes and renders them.
20. NVC invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.
21. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile which has both audio and video encoder configurations.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send SetAudioEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.5 for Invalid RTP header definition.

8.2.2 NVT MEDIA STREAMING – JPEG/G.711 (RTP-Unicast/RTSP/HTTP/TCP)

Test Label: Real Time Viewing NVT JPEG/G.711 Audio&Video streaming using HTTP transport.

ONVIF Core Specification Coverage: 11.1.1.4 RTP/RTSP/HTTP/TCP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP, 11.2.1.2 RTSP over HTTP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF SUPPORTED (Audio)

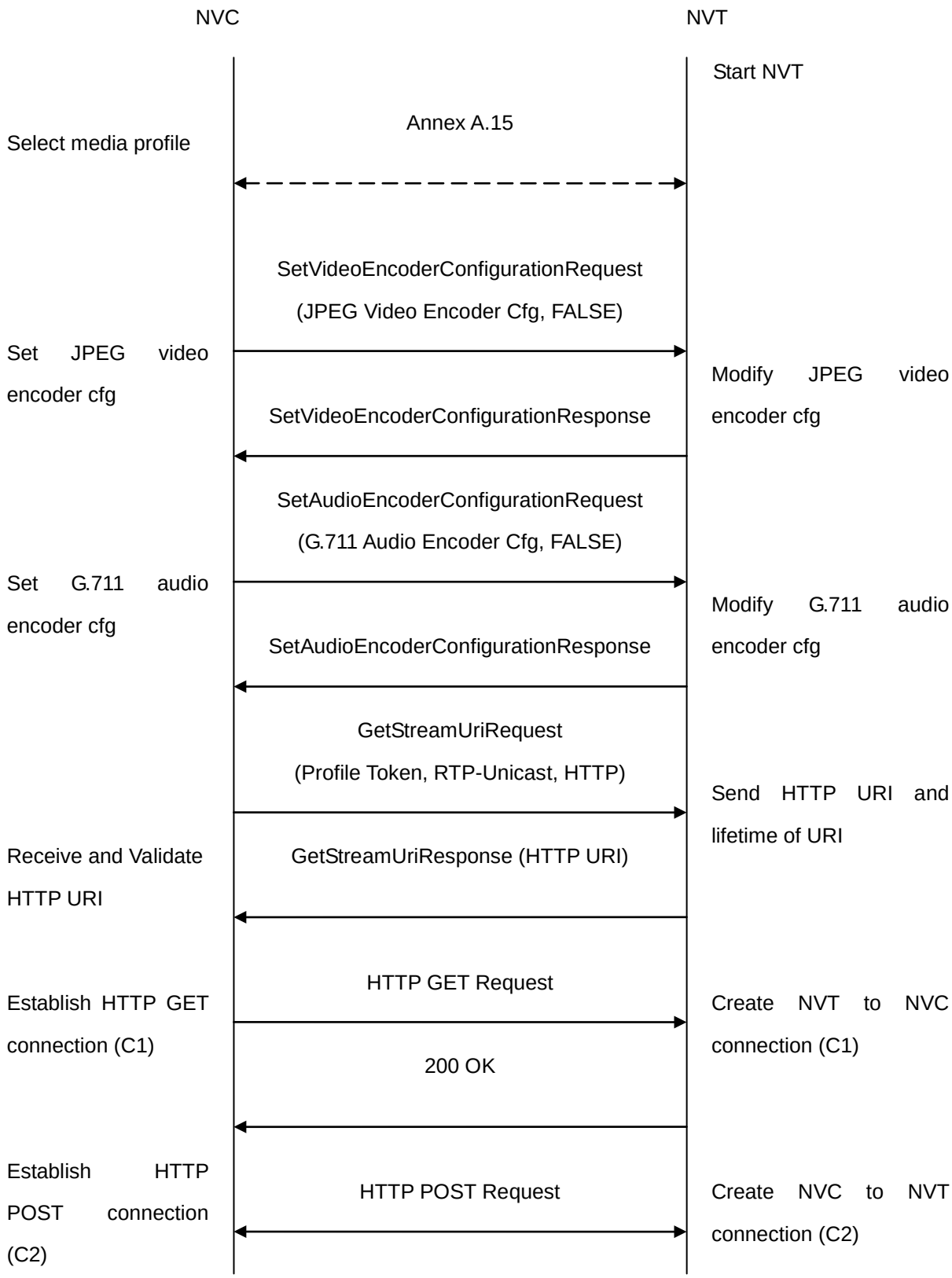
Test Purpose: To verify JPEG/G.711 Audio&Video streaming based on HTTP Transport.

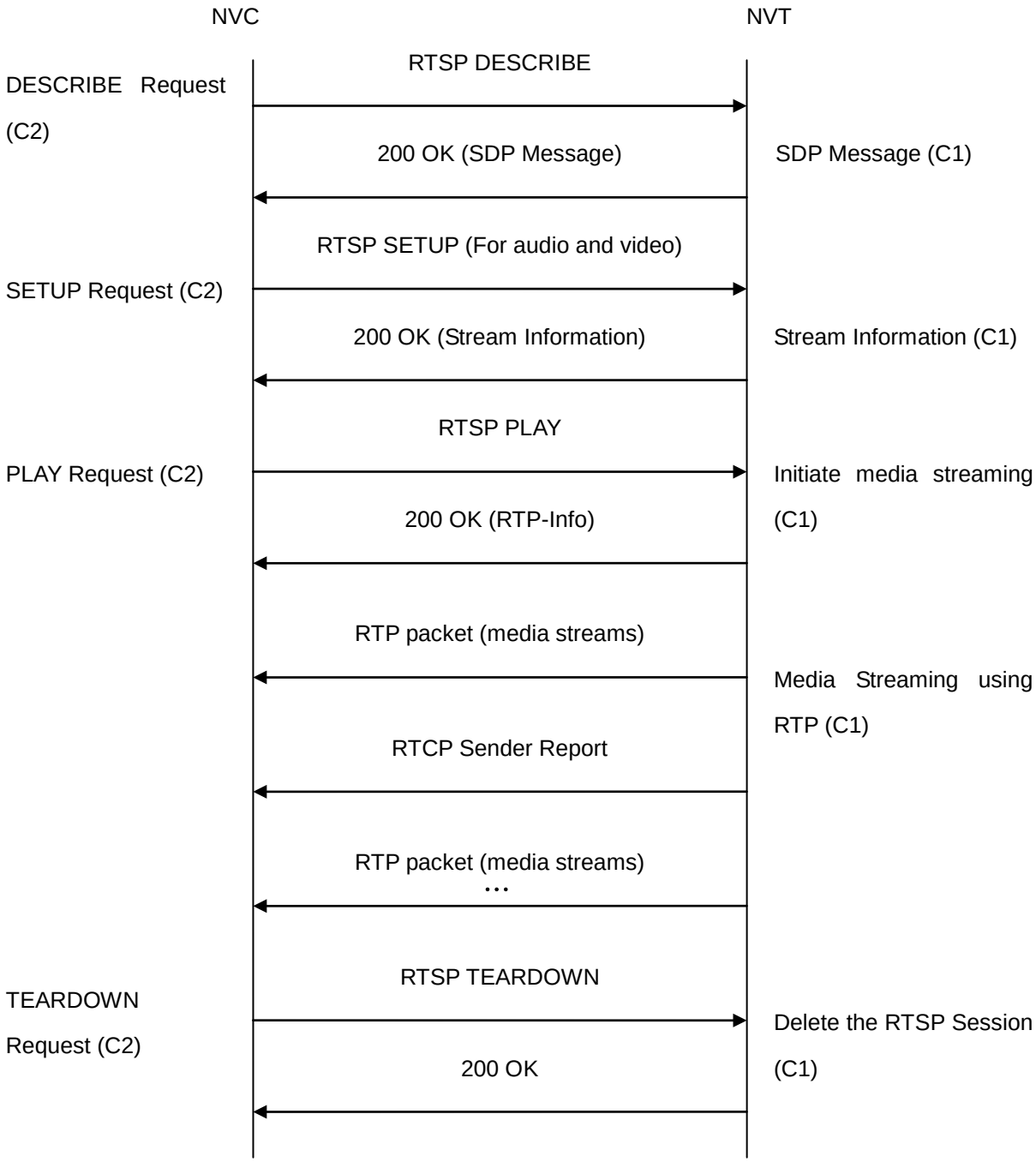
Pre-Requisite: Audio is supported by NVT.

A media profile with JPEG video encoder configuration and G.711 audio encoder configuration

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.15
Select media profile
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
Modify JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
SetAudioEncoderConfigurationRequest
(G.711 Audio Encoder Cfg, FALSE)
Set G.711 audio encoder cfg
Modify G.711 audio encoder cfg
SetAudioEncoderConfigurationResponse
GetStreamUriRequest
(Profile Token, RTP-Unicast, HTTP)
Send HTTP URI and lifetime of URI
Receive and Validate HTTP URI
GetStreamUriResponse (HTTP URI)
HTTP GET Request
Establish HTTP GET connection (C1)
Create NVT to NVC connection (C1)
200 OK
Establish HTTP POST connection (C2)
HTTP POST Request
Create NVC to NVT connection (C2)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with both JPEG video encoding support and G.711 audio encoding support by following the procedure mentioned in Annex A.15.

4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.15.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes SetAudioEncoderConfigurationRequest (**Encoding = "G711", Bitrate = r1, SampleRate = r2, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetAudioEncoderConfigurationOptions response in A.15.
7. NVT modifies audio encoder configuration and responds with SetAudioEncoderConfigurationResponse message indicating success.
8. NVC invokes GetStreamUriRequest message (Profile Token, RTP-Unicast, HTTP transport) to retrieve media stream URI for the selected media profile.
9. NVT sends HTTP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
10. NVC verifies the HTTP media stream URI provided by the NVT.
11. NVC invokes HTTP GET Request on NVT and establishes NVT to NVC connection for RTP data transfer.
12. NVC invokes HTTP POST Request and establishes NVC to NVT connection for RTSP control requests.
13. NVC invokes RTSP DESCRIBE request on HTTP POST connection.
14. NVT sends 200 OK message and SDP information on HTTP GET connection.
15. NVC invokes RTSP SETUP requests on HTTP POST connection with transport parameter as 'RTP/TCP' along with 'interleaved' parameter for both audio and video streams separately.
16. NVT sends 200 OK message and the media stream information on HTTP GET connection.
17. NVC invokes RTSP PLAY request on HTTP POST connection.
18. NVT sends 200 OK message and starts media streaming on HTTP GET connection.
19. NVT transfers Audio/Video RTP media stream to NVC on HTTP GET connection.
20. NVT sends Audio/Video RTCP sender report to NVC on HTTP GET connection.
21. NVT validates the received RTP and RTCP packets, decodes and renders them.
22. NVC invokes RTSP TEARDOWN control request on HTTP POST connection and closes the HTTP POST connection.
23. NVT sends 200 OK Response on HTTP GET connection and closes the HTTP GET connection.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile which has both audio and video encoder configurations.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send SetAudioEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.5 for Invalid RTP header definition.

8.2.3 NVT MEDIA STREAMING – JPEG/G.711 (RTP/RTSP/TCP)

Test Label: Real Time Viewing NVT JPEG/G.711 Audio&Video streaming using RTP/RTSP/TCP transport.

ONVIF Core Specification Coverage: 11.1.1.3 RTP/RTSP/TCP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF SUPPORTED (Audio) & IMPLEMENTED (RTP/RTSP/TCP)

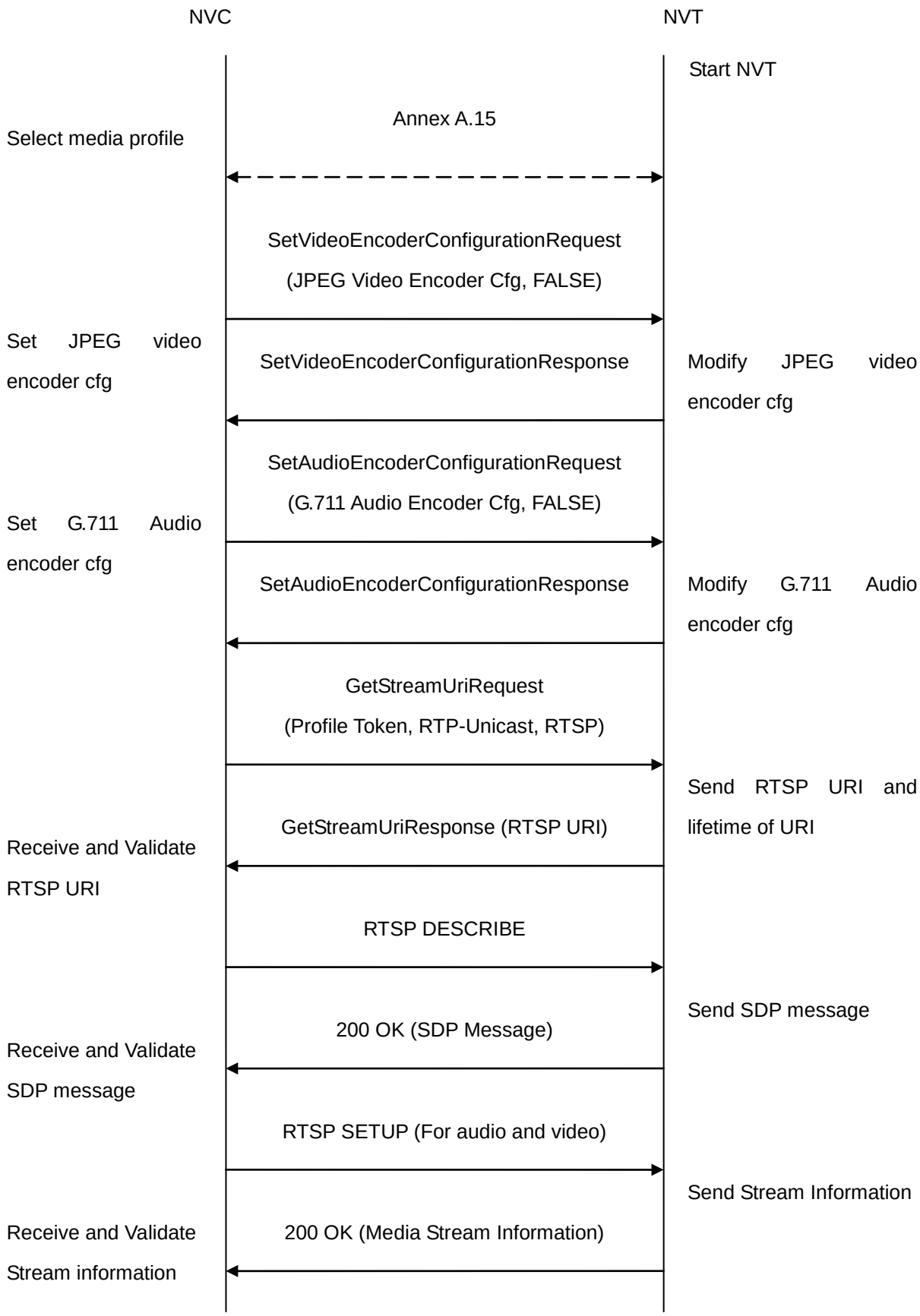
Test Purpose: To verify JPEG/G.711 Audio&Video streaming based on RTP/RTSP/TCP using RTSP tunnel.

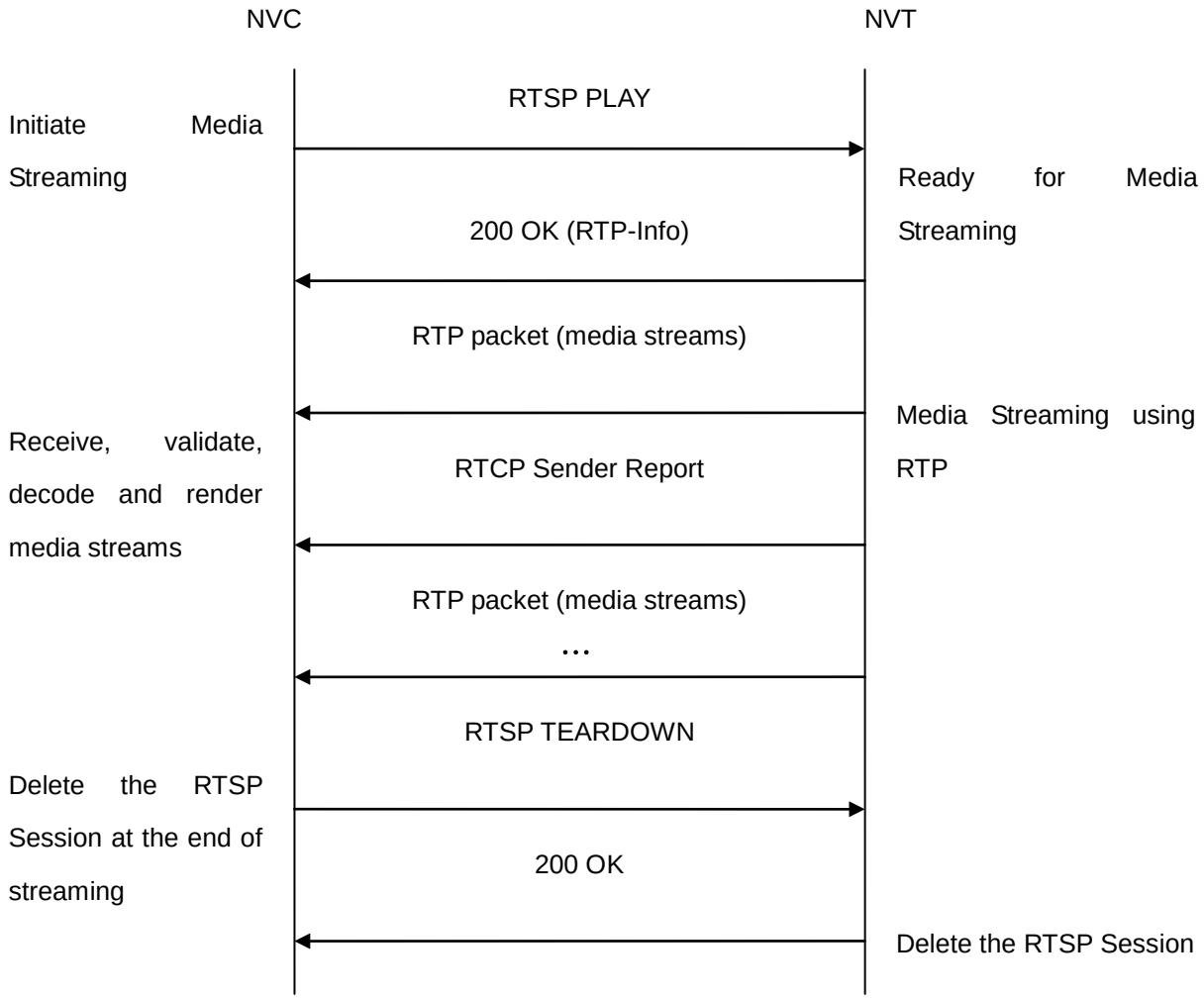
Pre-Requisite: Audio is supported by NVT and RTP/RTSP/TCP media streaming is implemented by NVT.

A media profile with JPEG video encoder configuration and G.711 audio encoder configuration

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.15
Select media profile
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
Modify JPEG video encoder cfg
SetAudioEncoderConfigurationRequest
(G.711 Audio Encoder Cfg, FALSE)
Set G.711 Audio encoder cfg
SetAudioEncoderConfigurationResponse
Modify G.711 Audio encoder cfg
GetStreamUriRequest
(Profile Token, RTP-Unicast, RTSP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Send SDP message
200 OK (SDP Message)
Receive and Validate SDP message
RTSP SETUP (For audio and video)
Send Stream Information
200 OK (Media Stream Information)
Receive and Validate Stream information



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with both JPEG video encoding support and G.711 audio encoding support by following the procedure mentioned in Annex A.15.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.15.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes SetAudioEncoderConfigurationRequest (**Encoding = "G711", Bitrate = r1, SampleRate = r2, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetAudioEncoderConfigurationOptions response in A.15.
7. NVT modifies audio encoder configuration and responds with SetAudioEncoderConfigurationResponse message indicating success.

8. NVC invokes GetStreamUriRequest message (Profile Token, RTP-Unicast, RTSP transport) to retrieve media stream URI for the selected media profile.
9. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
10. NVC verifies the RTSP media stream URI provided by the NVT.
11. NVC invokes RTSP DESCRIBE request.
12. NVT sends 200 OK message and SDP information.
13. NVC invokes RTSP SETUP request with transport parameter as 'RTP/TCP' along with 'interleaved' parameter for both audio and video streams separately.
14. NVT sends 200 OK message and the media stream information.
15. NVC invokes RTSP PLAY request.
16. NVT sends 200 OK message and starts media streaming.
17. NVT interleaves RTP and RTCP packets, send them over RTSP control connection.
18. NVT validates the received RTP and RTCP packets, decodes and renders them.
19. NVC invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.
20. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile which has both audio and video encoder configurations.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send SetAudioEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

8.2.4 NVT MEDIA STREAMING – JPEG/G.726 (RTP-Unicast/ UDP)

Test Label: Real Time Viewing NVT JPEG/G.726 Audio&Video streaming using RTP-Unicast/UDP transport.

ONVIF Core Specification Coverage: 11.1.1.1 RTP data transfer via UDP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF SUPPORTED (Audio) & IMPLEMENTED (G.726)

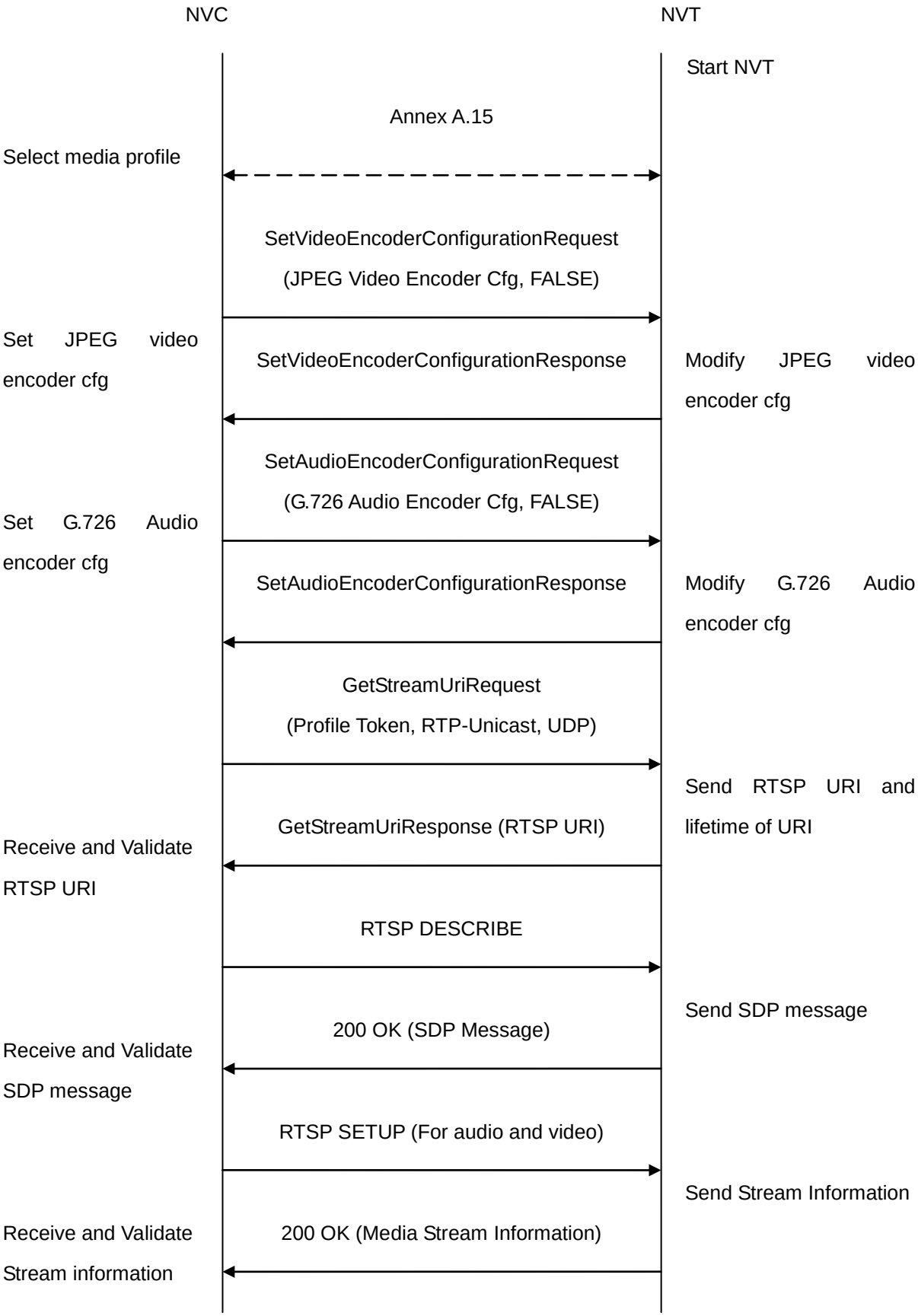
Test Purpose: To verify JPEG/G.726 Audio&Video streaming based on RTP/UDP Unicast Transport.

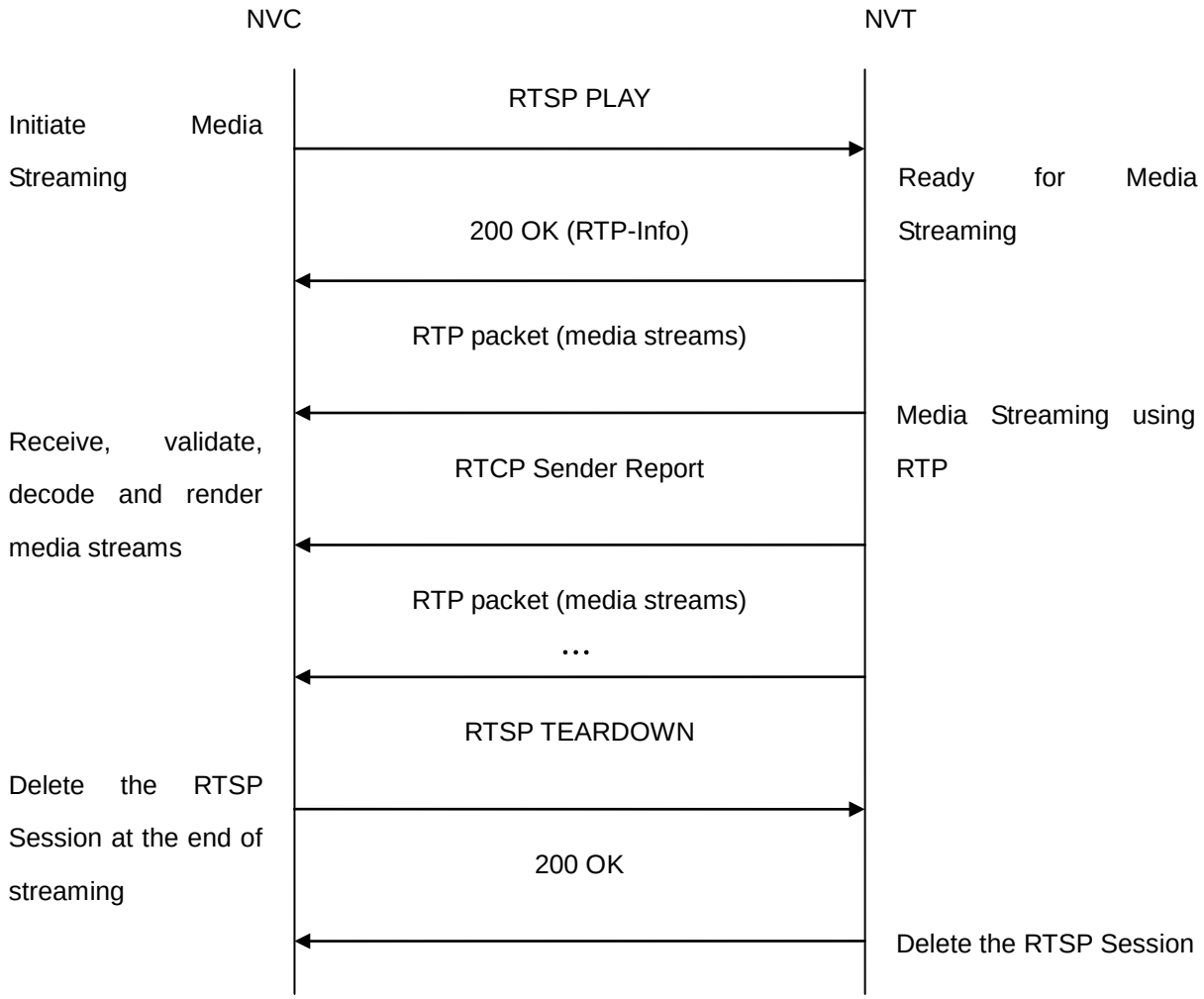
Pre-Requisite: Audio is supported by NVT and G.726 is implemented by NVT.

A media profile with JPEG video encoder configuration and G.726 audio encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.15
Select media profile
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
Modify JPEG video encoder cfg
SetAudioEncoderConfigurationRequest
(G.726 Audio Encoder Cfg, FALSE)
Set G.726 Audio encoder cfg
SetAudioEncoderConfigurationResponse
Modify G.726 Audio encoder cfg
GetStreamUriRequest
(Profile Token, RTP-Unicast, UDP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Send SDP message
200 OK (SDP Message)
Receive and Validate SDP message
RTSP SETUP (For audio and video)
Send Stream Information
200 OK (Media Stream Information)
Receive and Validate Stream information



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with both JPEG video encoding support and G.726 audio encoding support by following the procedure mentioned in Annex A.15.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.15.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes SetAudioEncoderConfigurationRequest (**Encoding = "G726", Bitrate = r1, SampleRate = r2, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetAudioEncoderConfigurationOptions response in A.15.
7. NVT modifies audio encoder configuration and responds with SetAudioEncoderConfigurationResponse message indicating success.

8. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, UDP transport**) to retrieve media stream URI for the selected media profile.
9. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
10. NVC verifies the RTSP media stream URI provided by the NVT.
11. NVC invokes RTSP DESCRIBE request.
12. NVT sends 200 OK message and SDP information.
13. NVC invokes RTSP SETUP request with transport parameter as **RTP/UDP** for both audio and video streams separately.
14. NVT sends 200 OK message and the media stream information.
15. NVC invokes RTSP PLAY request.
16. NVT sends 200 OK message and starts media streaming.
17. NVT sends Audio/Video RTP media stream to NVC over UDP.
18. NVT sends Audio/Video RTCP sender report to NVC.
19. NVT validates the received RTP and RTCP packets, decodes and renders them.
20. NVC invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.
21. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile which has both audio and video encoder configurations.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send SetAudioEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.5 for Invalid RTP header definition

8.2.5 NVT MEDIA STREAMING – JPEG/G.726 (RTP-Unicast/RTSP/HTTP/TCP)

Test Label: Real Time Viewing NVT JPEG/G.726 Audio&Video streaming using HTTP transport.

ONVIF Core Specification Coverage: 11.1.1.4 RTP/RTSP/HTTP/TCP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP, 11.2.1.2 RTSP over HTTP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF SUPPORTED (Audio) & IMPLEMENTED (G.726)

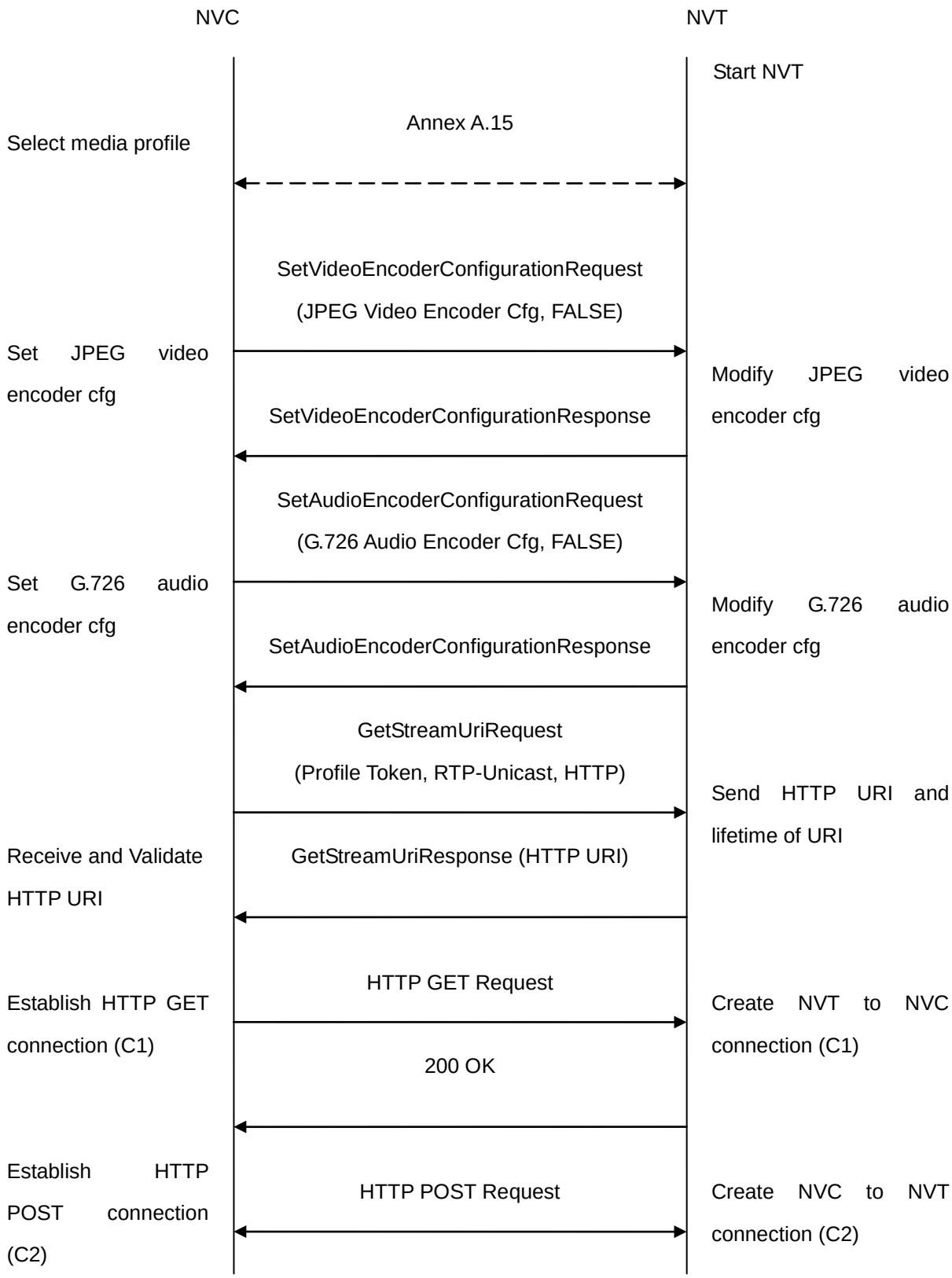
Test Purpose: To verify JPEG/G.726 Audio&Video streaming based on HTTP Transport.

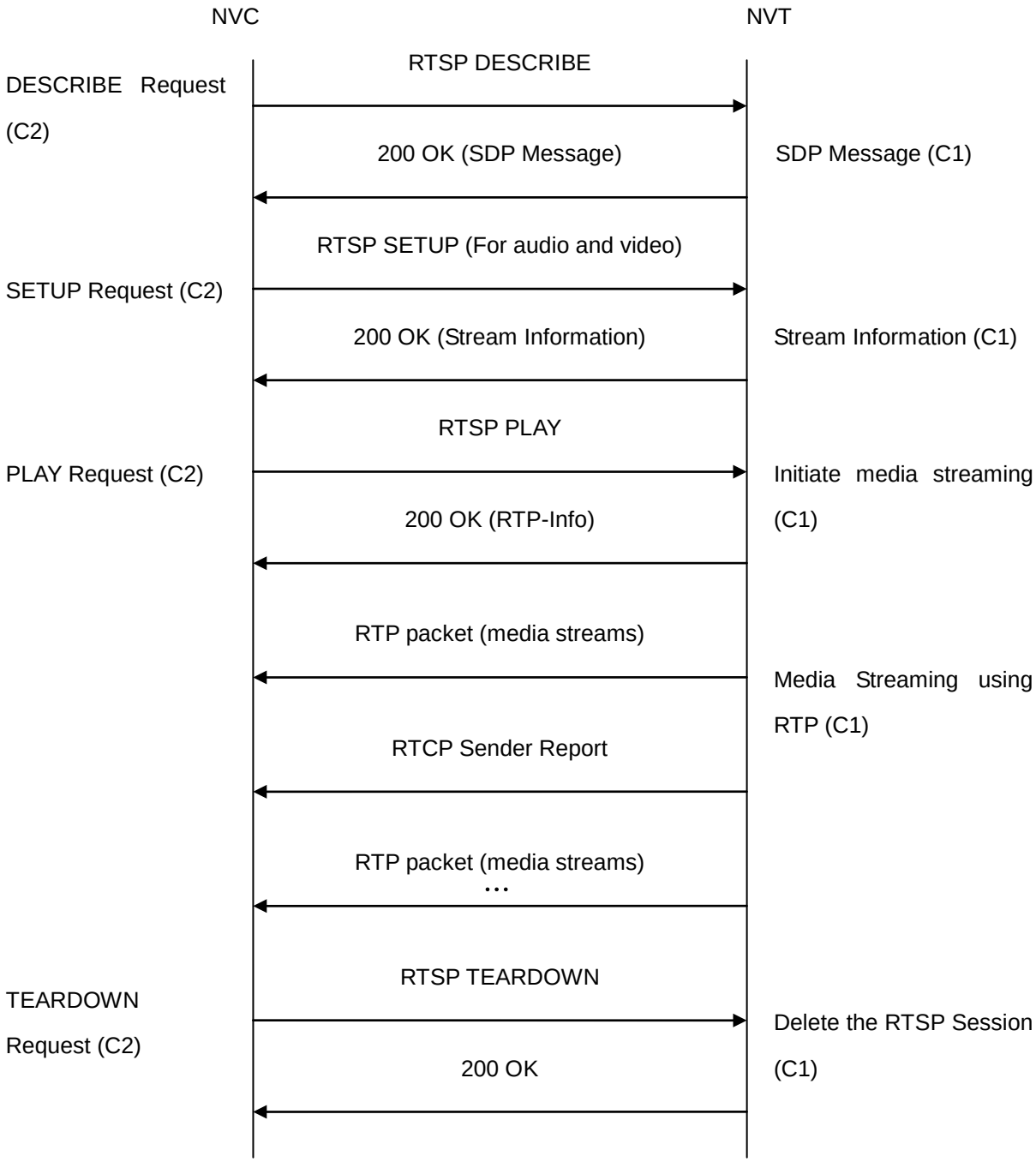
Pre-Requisite: Audio is supported by NVT and G.726 is implemented by NVT.

A media profile with JPEG video encoder configuration and G.726 audio encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.15
Select media profile
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
Modify JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
SetAudioEncoderConfigurationRequest
(G.726 Audio Encoder Cfg, FALSE)
Set G.726 audio encoder cfg
Modify G.726 audio encoder cfg
SetAudioEncoderConfigurationResponse
GetStreamUriRequest
(Profile Token, RTP-Unicast, HTTP)
Send HTTP URI and lifetime of URI
Receive and Validate HTTP URI
GetStreamUriResponse (HTTP URI)
HTTP GET Request
Establish HTTP GET connection (C1)
Create NVT to NVC connection (C1)
200 OK
Establish HTTP POST connection (C2)
HTTP POST Request
Create NVC to NVT connection (C2)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with both JPEG video encoding support and G.726 audio encoding support by following the procedure mentioned in Annex A.15.

4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.15.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes SetAudioEncoderConfigurationRequest (**Encoding = "G726", Bitrate = r1, SampleRate = r2, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetAudioEncoderConfigurationOptions response in A.15.
7. NVT modifies audio encoder configuration and responds with SetAudioEncoderConfigurationResponse message indicating success.
8. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, HTTP transport**) to retrieve media stream URI for the selected media profile.
9. NVT sends **HTTP URI** and parameters defining the lifetime of the URI like **ValidUntilConnect, ValidUntilReboot and Timeout** in the GetStreamUriResponse message.
10. NVC verifies the HTTP media stream URI provided by the NVT.
11. NVC invokes HTTP GET Request on NVT and establishes NVT to NVC connection for RTP data transfer.
12. NVC invokes HTTP POST Request and establishes NVC to NVT connection for RTSP control requests.
13. NVC invokes RTSP DESCRIBE request on HTTP POST connection.
14. NVT sends 200 OK message and SDP information on HTTP GET connection.
15. NVC invokes RTSP SETUP requests on HTTP POST connection with transport parameter as **'RTP/TCP'** along with **'interleaved'** parameter for both audio and video streams separately.
16. NVT sends 200 OK message and the media stream information on HTTP GET connection.
17. NVC invokes RTSP PLAY request on HTTP POST connection.
18. NVT sends 200 OK message and starts media streaming on HTTP GET connection.
19. NVT transfers Audio/Video RTP media stream to NVC on HTTP GET connection.
20. NVT sends Audio/Video RTCP sender report to NVC on HTTP GET connection.
21. NVT validates the received RTP and RTCP packets, decodes and renders them.
22. NVC invokes RTSP TEARDOWN control request on HTTP POST connection and closes the HTTP POST connection.
23. NVT sends 200 OK Response on HTTP GET connection and closes the HTTP GET connection.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile which has both audio and video encoder configurations.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send SetAudioEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.5 for Invalid RTP header definition.

8.2.6 NVT MEDIA STREAMING – JPEG/G.726 (RTP/RTSP/TCP)

Test Label: Real Time Viewing NVT JPEG/G.726 Audio&Video streaming using RTP/RTSP/TCP transport.

ONVIF Core Specification Coverage: 11.1.1.3 RTP/RTSP/TCP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF SUPPORTED (Audio) & IMPLEMENTED (G.726 & RTP/RTSP/TCP)

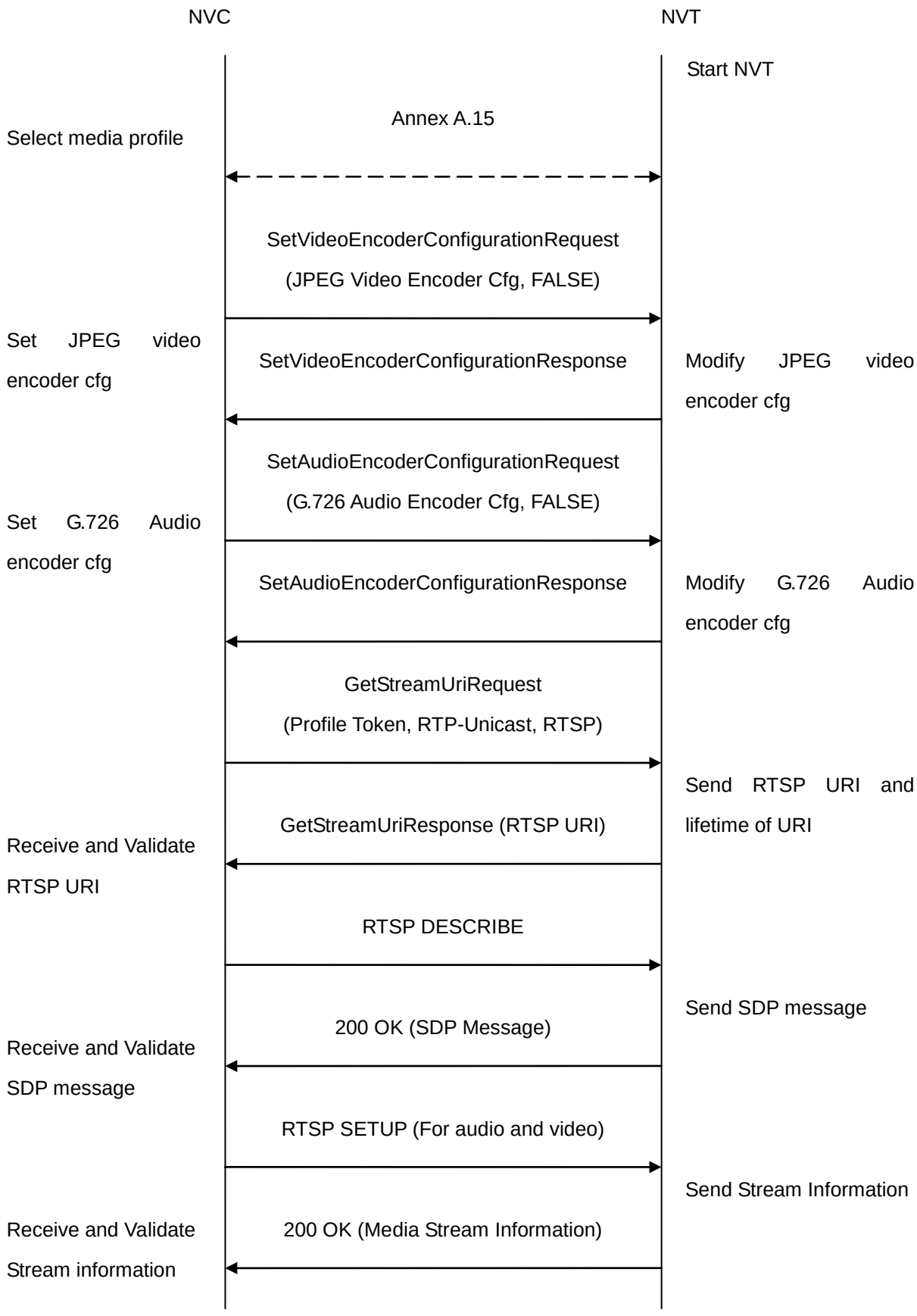
Test Purpose: To verify JPEG/G.726 Audio&Video streaming based on RTP/RTSP/TCP using RTSP tunnel.

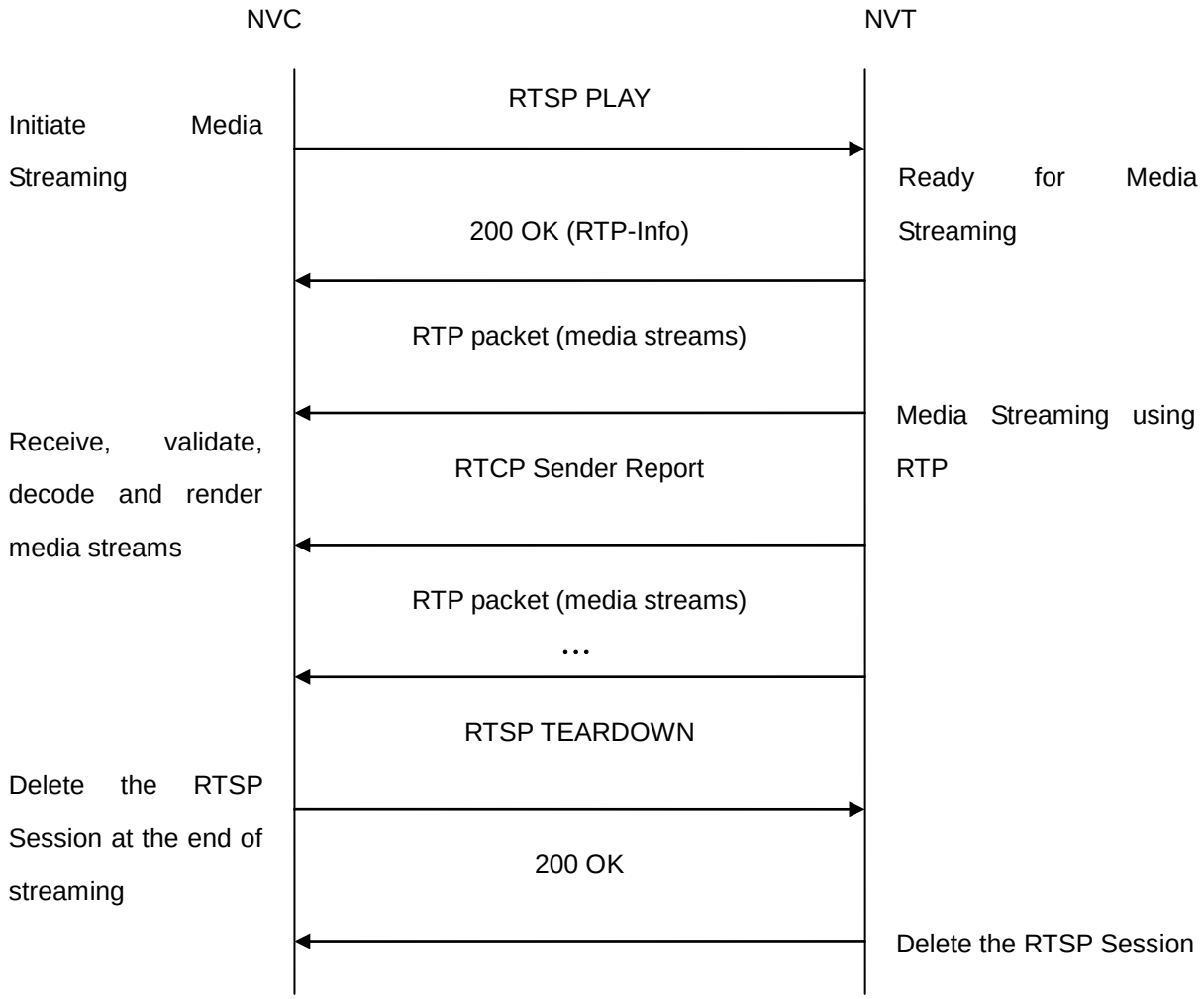
Pre-Requisite: Audio is supported by NVT, G.726 and RTP/RTSP/TCP media streaming is implemented by NVT.

A media profile with JPEG video encoder configuration and G.726 audio encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.15
Select media profile
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
Modify JPEG video encoder cfg
SetAudioEncoderConfigurationRequest
(G.726 Audio Encoder Cfg, FALSE)
Set G.726 Audio encoder cfg
SetAudioEncoderConfigurationResponse
Modify G.726 Audio encoder cfg
GetStreamUriRequest
(Profile Token, RTP-Unicast, RTSP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Send SDP message
200 OK (SDP Message)
Receive and Validate SDP message
RTSP SETUP (For audio and video)
Send Stream Information
200 OK (Media Stream Information)
Receive and Validate Stream information



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with both JPEG video encoding support and G.726 audio encoding support by following the procedure mentioned in Annex A.15.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.15.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes SetAudioEncoderConfigurationRequest (**Encoding = "G726", Bitrate = r1, SampleRate = r2, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetAudioEncoderConfigurationOptions response in A.15.
7. NVT modifies audio encoder configuration and responds with SetAudioEncoderConfigurationResponse message indicating success.

8. NVC invokes GetStreamUri request (**Profile Token, RTP-Unicast, RTSP transport**) to retrieve media stream URI for the selected media profile.
9. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
10. NVC verifies the RTSP media stream URI provided by the NVT.
11. NVC invokes RTSP DESCRIBE request.
12. NVT sends 200 OK message and SDP information.
13. NVC invokes RTSP SETUP request with transport parameter as **'RTP/TCP'** along with **'interleaved'** parameter for both audio and video streams separately.
14. NVT sends 200 OK message and the media stream information.
15. NVC invokes RTSP PLAY request.
16. NVT sends 200 OK message and starts media streaming.
17. NVT interleaves RTP and RTCP packets, send them over RTSP control connection.
18. NVT validates the received RTP and RTCP packets, decodes and renders them.
19. NVC invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.
20. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile which has both audio and video encoder configurations.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send SetAudioEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

8.2.7 NVT MEDIA STREAMING – JPEG/AAC (RTP-Unicast/ UDP)

Test Label: Real Time Viewing NVT JPEG/AAC Audio&Video streaming using RTP-Unicast/UDP transport.

ONVIF Core Specification Coverage: 11.1.1.1 RTP data transfer via UDP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF SUPPORTED (Audio) & IMPLEMENTED (AAC)

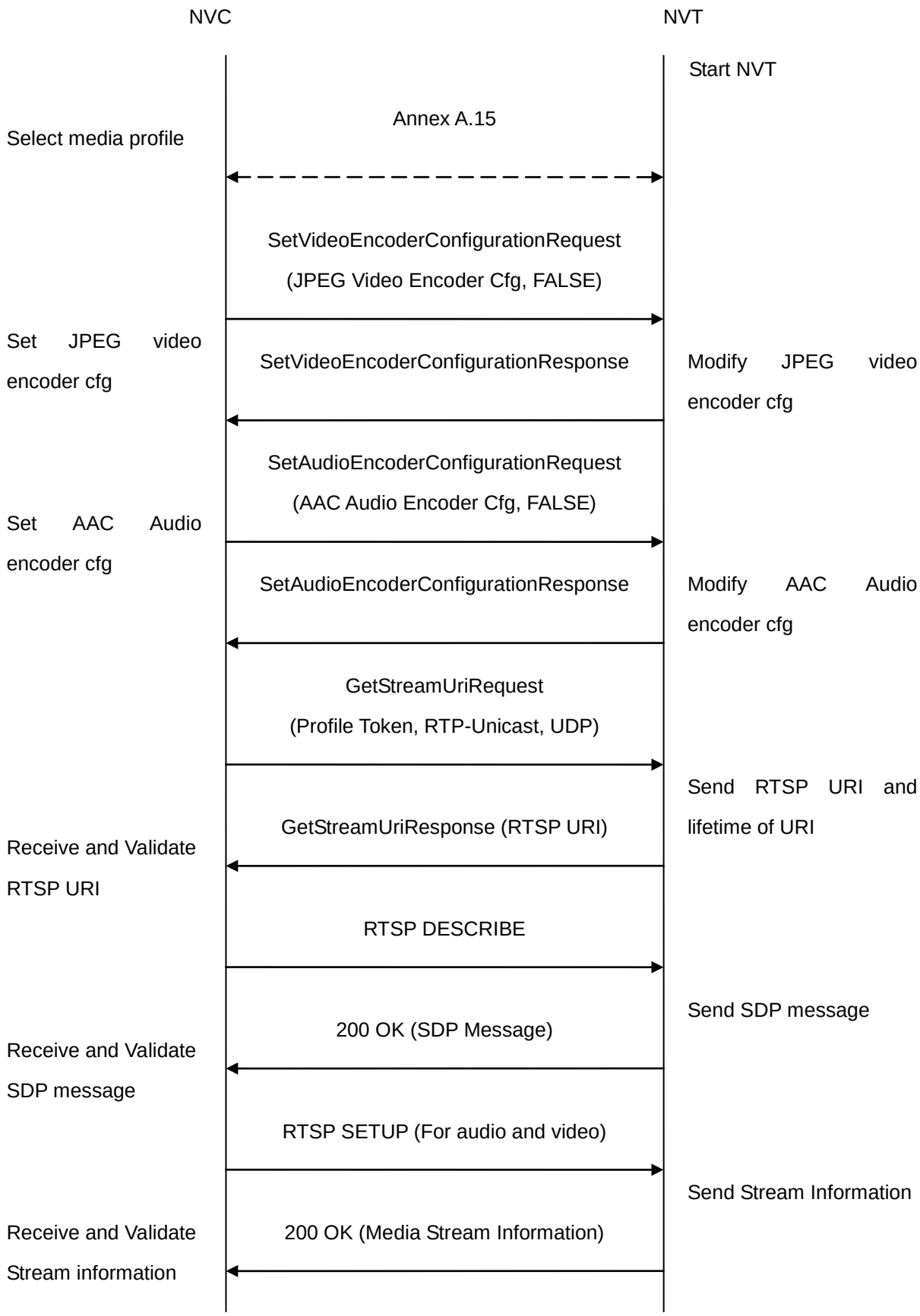
Test Purpose: To verify JPEG/AAC Audio&Video streaming based on RTP/UDP Unicast Transport.

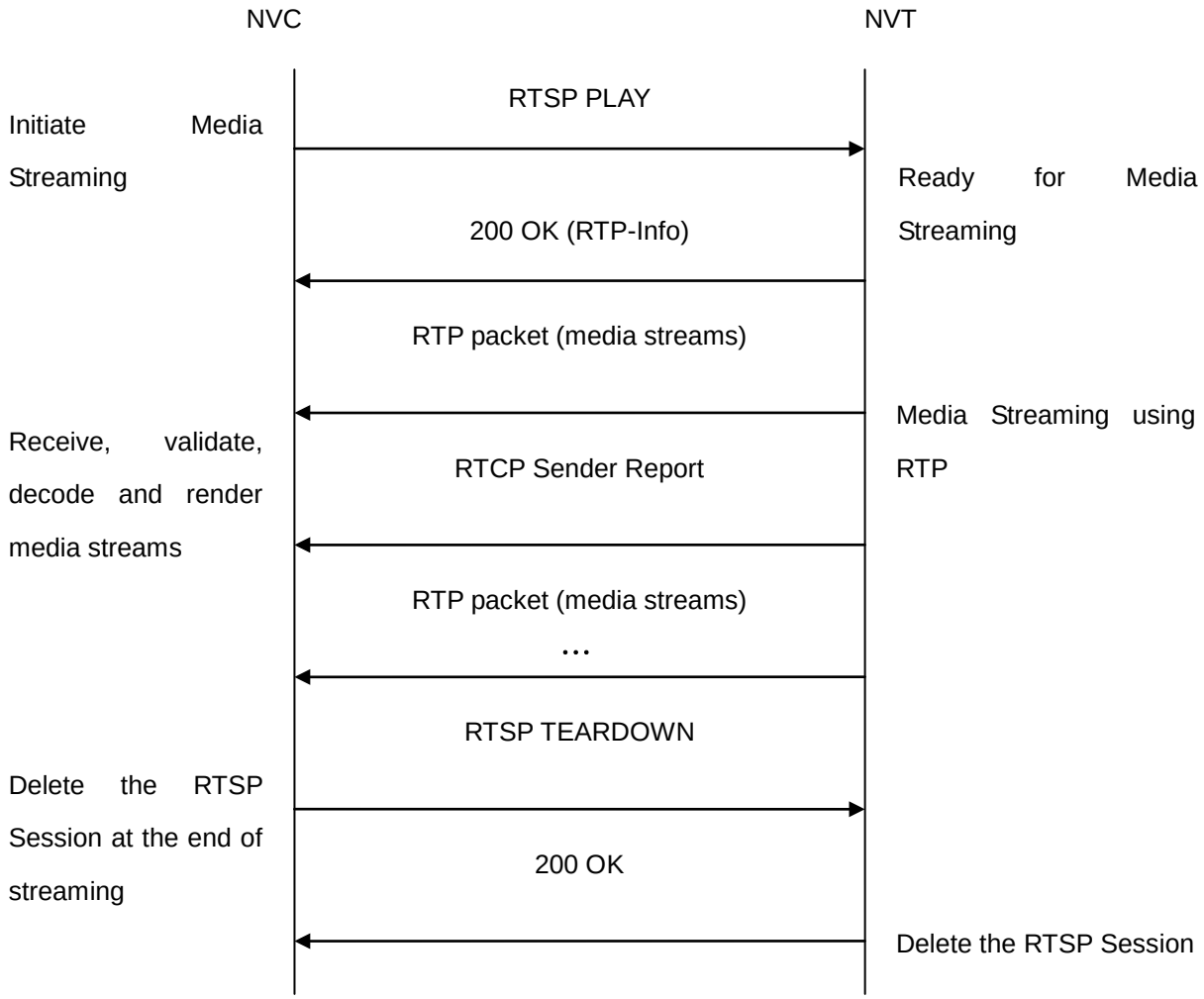
Pre-Requisite: Audio is supported by NVT and AAC is implemented by NVT.

A media profile with JPEG video encoder configuration and AAC audio encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.15
Select media profile
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
Modify JPEG video encoder cfg
SetAudioEncoderConfigurationRequest
(AAC Audio Encoder Cfg, FALSE)
Set AAC Audio encoder cfg
SetAudioEncoderConfigurationResponse
Modify AAC Audio encoder cfg
GetStreamUriRequest
(Profile Token, RTP-Unicast, UDP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Send SDP message
200 OK (SDP Message)
Receive and Validate SDP message
RTSP SETUP (For audio and video)
Send Stream Information
200 OK (Media Stream Information)
Receive and Validate Stream information



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with both JPEG video encoding support and AAC audio encoding support by following the procedure mentioned in Annex A.15.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.15.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes SetAudioEncoderConfigurationRequest (**Encoding = "AAC", Bitrate = r1, SampleRate = r2, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetAudioEncoderConfigurationOptions response in A.15.
7. NVT modifies audio encoder configuration and responds with SetAudioEncoderConfigurationResponse message indicating success.

8. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, UDP transport**) to retrieve media stream URI for the selected media profile.
9. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
10. NVC verifies the RTSP media stream URI provided by the NVT.
11. NVC invokes RTSP DESCRIBE request.
12. NVT sends 200 OK message and SDP information.
13. NVC invokes RTSP SETUP requests for audio and video streams separately with transport parameter as **RTP/UDP** for both audio and video streams separately.
14. NVT sends 200 OK message and the media stream information.
15. NVC invokes RTSP PLAY request.
16. NVT sends 200 OK message and starts media streaming.
17. NVT sends Audio/Video RTP media stream to NVC over UDP.
18. NVT sends Audio/Video RTCP sender report to NVC.
19. NVT validates the received RTP and RTCP packets, decodes and renders them.
20. NVC invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.
21. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile which has both audio and video encoder configurations.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send SetAudioEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.5 for Invalid RTP header definition.

8.2.8 NVT MEDIA STREAMING – JPEG/AAC (RTP-Unicast/RTSP/HTTP/TCP)

Test Label: Real Time Viewing NVT JPEG/AAC Audio&Video streaming using HTTP transport.

ONVIF Core Specification Coverage: 11.1.1.4 RTP/RTSP/HTTP/TCP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP, 11.2.1.2 RTSP over HTTP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF SUPPORTED (Audio) & IMPLEMENTED (AAC)

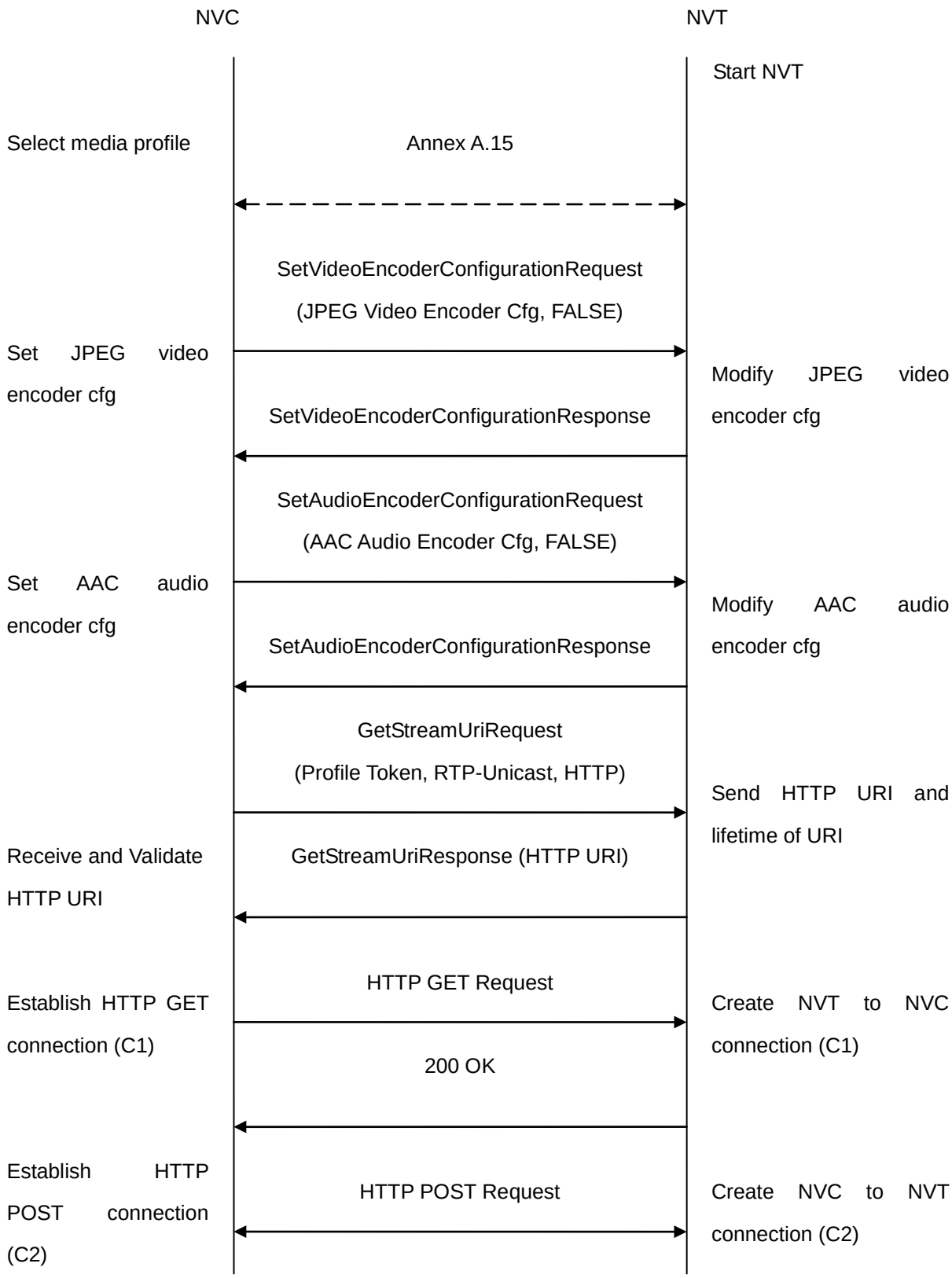
Test Purpose: To verify JPEG/AAC Audio&Video streaming based on HTTP Transport.

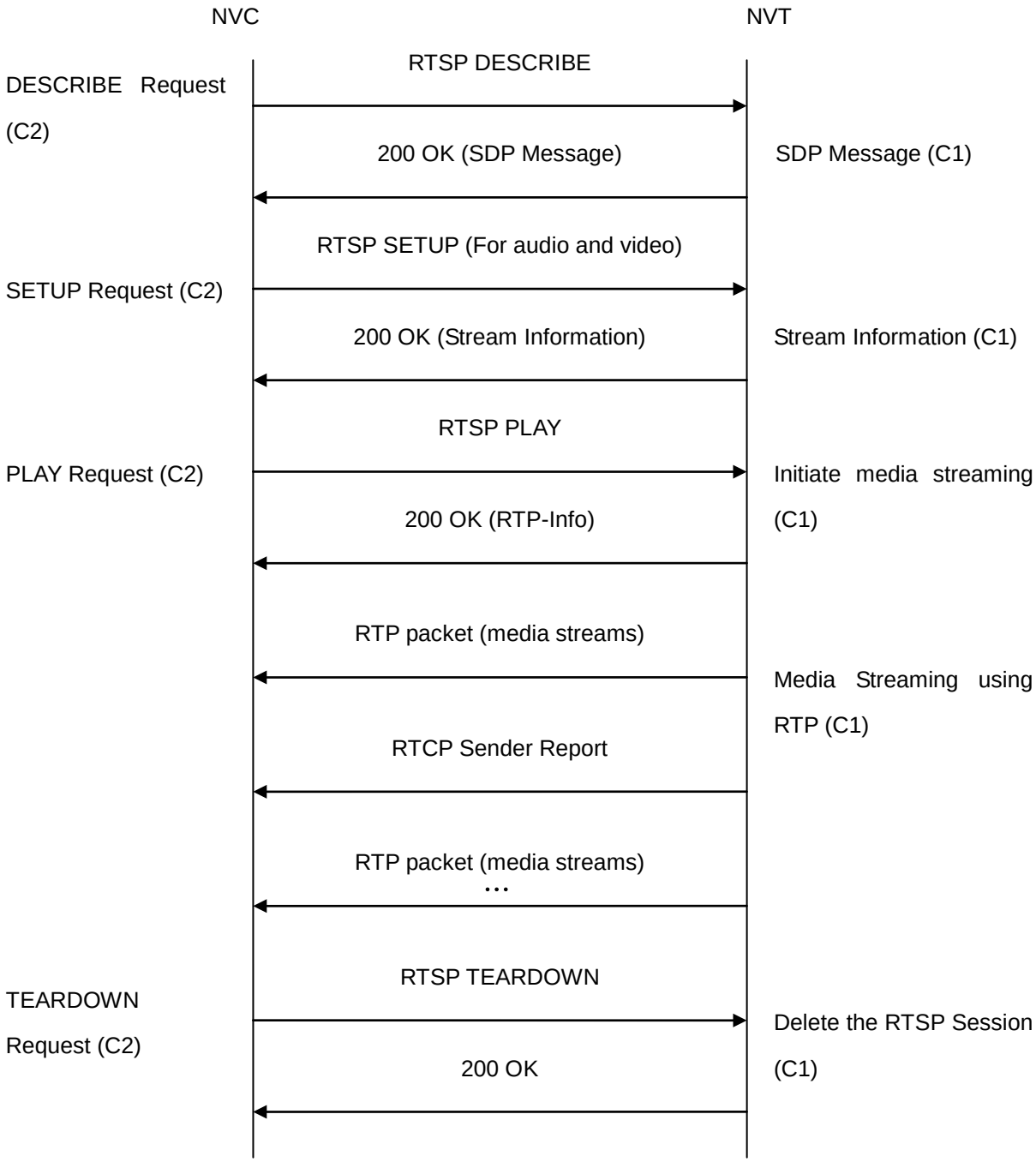
Pre-Requisite: Audio is supported by NVT and AAC is implemented by NVT.

A media profile with JPEG video encoder configuration and AAC audio encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Select media profile
Annex A.15
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
Modify JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
SetAudioEncoderConfigurationRequest
(AAC Audio Encoder Cfg, FALSE)
Set AAC audio encoder cfg
Modify AAC audio encoder cfg
SetAudioEncoderConfigurationResponse
GetStreamUriRequest
(Profile Token, RTP-Unicast, HTTP)
Send HTTP URI and lifetime of URI
Receive and Validate HTTP URI
GetStreamUriResponse (HTTP URI)
HTTP GET Request
Establish HTTP GET connection (C1)
Create NVT to NVC connection (C1)
200 OK
Establish HTTP POST connection (C2)
HTTP POST Request
Create NVC to NVT connection (C2)



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with both JPEG video encoding support and AAC audio encoding support by following the procedure mentioned in Annex A.15.

4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.15.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes SetAudioEncoderConfigurationRequest (**Encoding = "AAC", Bitrate = r1, SampleRate = r2, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetAudioEncoderConfigurationOptions response in A.15.
7. NVT modifies audio encoder configuration and responds with SetAudioEncoderConfigurationResponse message indicating success.
8. NVC invokes GetStreamUriRequest message (**Profile Token, RTP-Unicast, HTTP transport**) to retrieve media stream URI for the selected media profile.
9. NVT sends HTTP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
10. NVC verifies the HTTP media stream URI provided by the NVT.
11. NVC invokes HTTP GET Request on NVT and establishes NVT to NVC connection for RTP data transfer.
12. NVC invokes HTTP POST Request and establishes NVC to NVT connection for RTSP control requests.
13. NVC invokes RTSP DESCRIBE request on HTTP POST connection.
14. NVT sends 200 OK message and SDP information on HTTP GET connection.
15. NVC invokes RTSP SETUP request on HTTP POST connection with transport parameter as **'RTP/TCP'** along with **'interleaved'** parameter for both audio and video streams separately.
16. NVT sends 200 OK message and the media stream information on HTTP GET connection.
17. NVC invokes RTSP PLAY request on HTTP POST connection.
18. NVT sends 200 OK message and starts media streaming on HTTP GET connection.
19. NVT transfers Audio/Video RTP media stream to NVC on HTTP GET connection.
20. NVT sends Audio/Video RTCP sender report to NVC on HTTP GET connection.
21. NVT validates the received RTP and RTCP packets, decodes and renders them.
22. NVC invokes RTSP TEARDOWN control request on HTTP POST connection and closes the HTTP POST connection.
23. NVT sends 200 OK Response on HTTP GET connection and closes the HTTP GET connection.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile which has both audio and video encoder configurations.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send SetAudioEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send valid RTP header in one or more media streams.

DUT did not send RTCP sender report correctly.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters.

See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

See Annex A.5 for Invalid RTP header definition.

8.2.9 NVT MEDIA STREAMING – JPEG/AAC (RTP/RTSP/TCP)

Test Label: Real Time Viewing NVT JPEG/AAC Audio&Video streaming using RTP/RTSP/TCP transport.

ONVIF Core Specification Coverage: 11.1.1.3 RTP/RTSP/TCP, 11.1.2.1 RTP, 11.1.2.2 RTCP, 11.2.1 Stream control, 11.2.1.1 RTSP.

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST IF SUPPORTED (Audio) & IMPLEMENTED (AAC & RTP/RTSP/TCP)

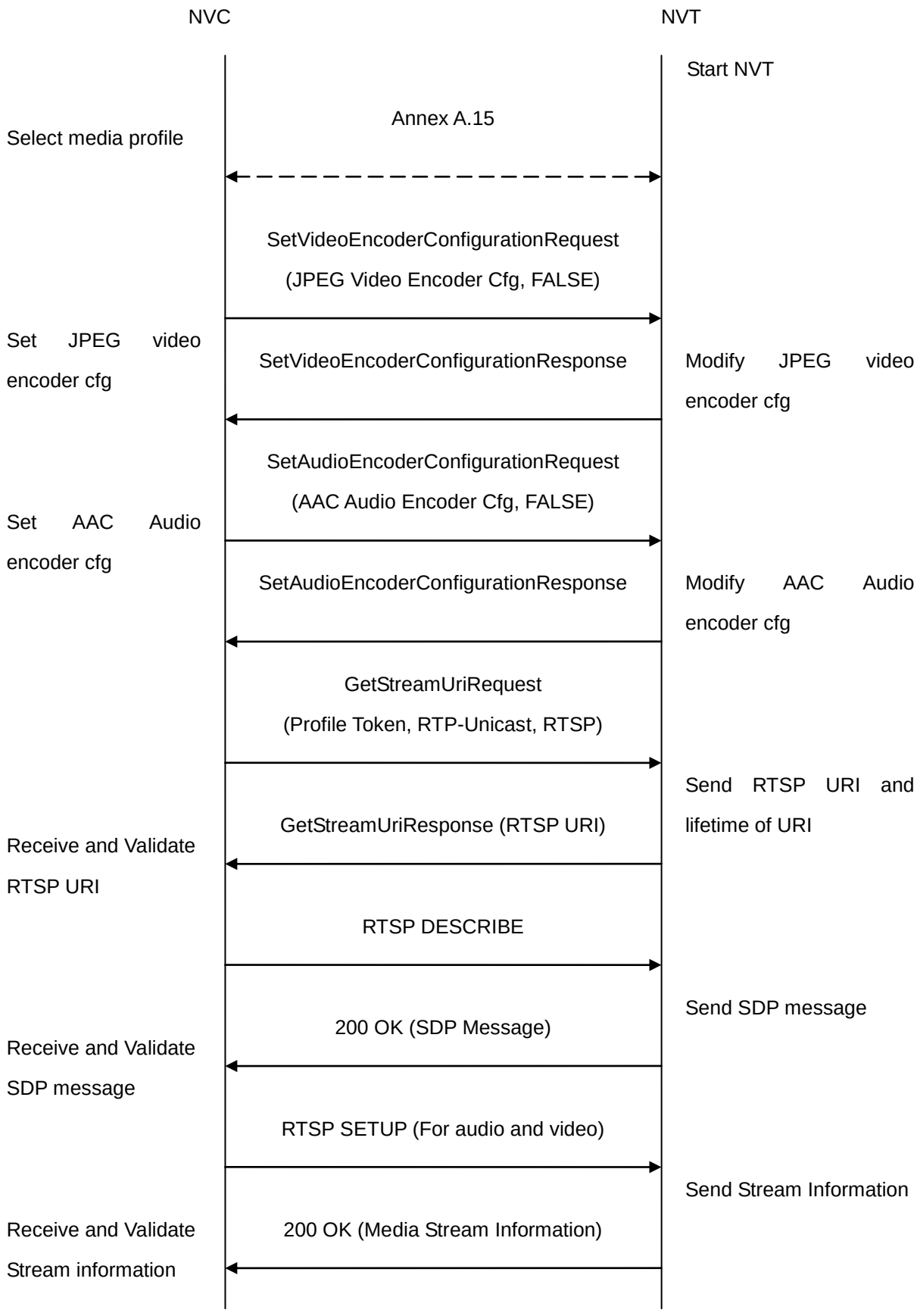
Test Purpose: To verify JPEG/AAC Audio&Video streaming based on RTP/RTSP/TCP using RTSP tunnel.

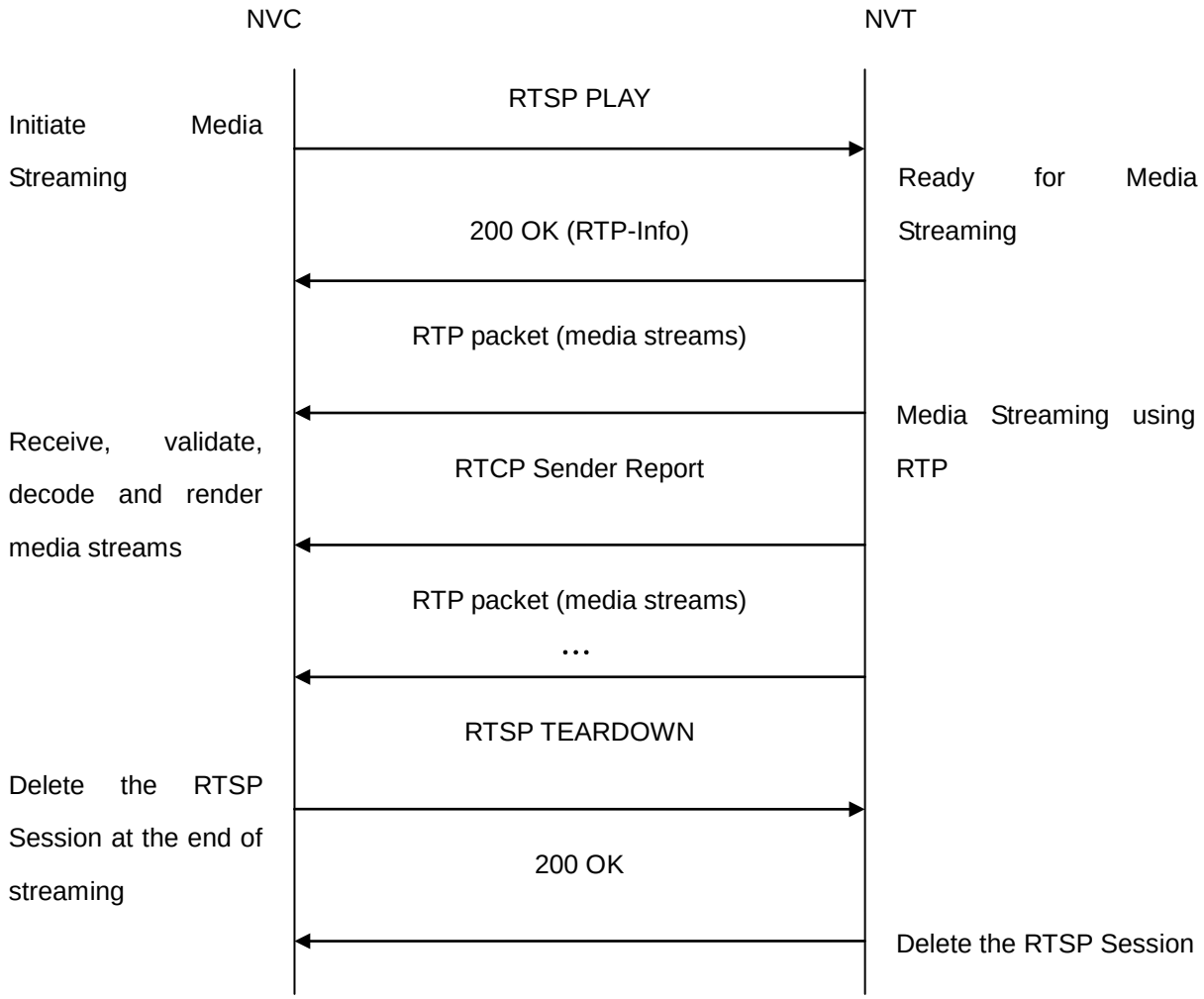
Pre-Requisite: Audio is supported by NVT, AAC and RTP/RTSP/TCP media streaming is implemented by NVT.

A media profile with JPEG video encoder configuration and AAC audio encoder configuration.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Start NVT
Annex A.15
Select media profile
SetVideoEncoderConfigurationRequest
(JPEG Video Encoder Cfg, FALSE)
Set JPEG video encoder cfg
SetVideoEncoderConfigurationResponse
Modify JPEG video encoder cfg
SetAudioEncoderConfigurationRequest
(AAC Audio Encoder Cfg, FALSE)
Set AAC Audio encoder cfg
SetAudioEncoderConfigurationResponse
Modify AAC Audio encoder cfg
GetStreamUriRequest
(Profile Token, RTP-Unicast, RTSP)
Send RTSP URI and lifetime of URI
GetStreamUriResponse (RTSP URI)
Receive and Validate RTSP URI
RTSP DESCRIBE
Send SDP message
200 OK (SDP Message)
Receive and Validate SDP message
RTSP SETUP (For audio and video)
Send Stream Information
200 OK (Media Stream Information)
Receive and Validate Stream information



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC selects a media profile with both JPEG video encoding support and AAC audio encoding support by following the procedure mentioned in Annex A.15.
4. NVC invokes SetVideoEncoderConfigurationRequest (**Encoding = "JPEG", Resolution = ["Width", "Height"], Quality = q1, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetVideoEncoderConfigurationOptions response in A.15.
5. NVT modifies video encoder configuration and responds with SetVideoEncoderConfigurationResponse message indicating success.
6. NVC invokes SetAudioEncoderConfigurationRequest (**Encoding = "AAC", Bitrate = r1, SampleRate = r2, Session Timeout = t1 and force persistence = false**). These values will be taken from the GetAudioEncoderConfigurationOptions response in A.15.
7. NVT modifies audio encoder configuration and responds with SetAudioEncoderConfigurationResponse message indicating success.

8. NVC invokes GetStreamUriRequest message (**Profile Token,RTP-Unicast,RTSP transport**) to retrieve media stream URI for the selected media profile.
9. NVT sends RTSP URI and parameters defining the lifetime of the URI like ValidUntilConnect, ValidUntilReboot and Timeout in the GetStreamUriResponse message.
10. NVC verifies the RTSP media stream URI provided by the NVT.
11. NVC invokes RTSP DESCRIBE request.
12. NVT sends 200 OK message and SDP information.
13. NVC invokes RTSP SETUP request with transport parameter as **'RTP/TCP'** along with **'interleaved'** parameter for both audio and video streams separately.
14. NVT sends 200 OK message and the media stream information.
15. NVC invokes RTSP PLAY request.
16. NVT sends 200 OK message and starts media streaming.
17. NVT interleaves RTP and RTCP packets, send them over RTSP control connection.
18. NVT validates the received RTP and RTCP packets, decodes and renders them.
19. NVC invokes RTSP TEARDOWN control request at the end of media streaming to terminate the RTSP session.
20. NVT sends 200 OK Response and terminates the RTSP Session.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not have valid media profile which has both audio and video encoder configurations.

DUT did not send SetVideoEncoderConfigurationResponse message.

DUT did not send SetAudioEncoderConfigurationResponse message.

DUT did not send GetStreamUriResponse message.

DUT did not send one or more mandatory parameters in the GetStreamUriResponse message (mandatory parameters – RTSP URI, ValidUntilConnect, ValidUntilReboot and Timeout).

DUT did not send RTSP 200 OK response for RTSP DESCRIBE, SETUP, PLAY and TEARDOWN requests.

DUT did not send RTP and RTCP packets as per [RFC 2326] section 10.12.

RTSP Session is terminated by DUT during media streaming.

Note: See Annex A.7 for usage of ValidUntilConnect, ValidUntilReboot, and Timeout parameters. See Annex A.10 for correct syntax for the StreamSetup element in GetStreamUri requests.

9 Event Handling Test Cases

9.1 Event Properties

9.1.1 NVT GET EVENT PROPERTIES

Test Label: Event handling GET EVENT PROPERTIES

ONVIF Core Specification Coverage: 12.8 GetEventProperties

Device Type: NVT

Command Under Test: GetEventProperties

WSDL Reference: event.wsdl

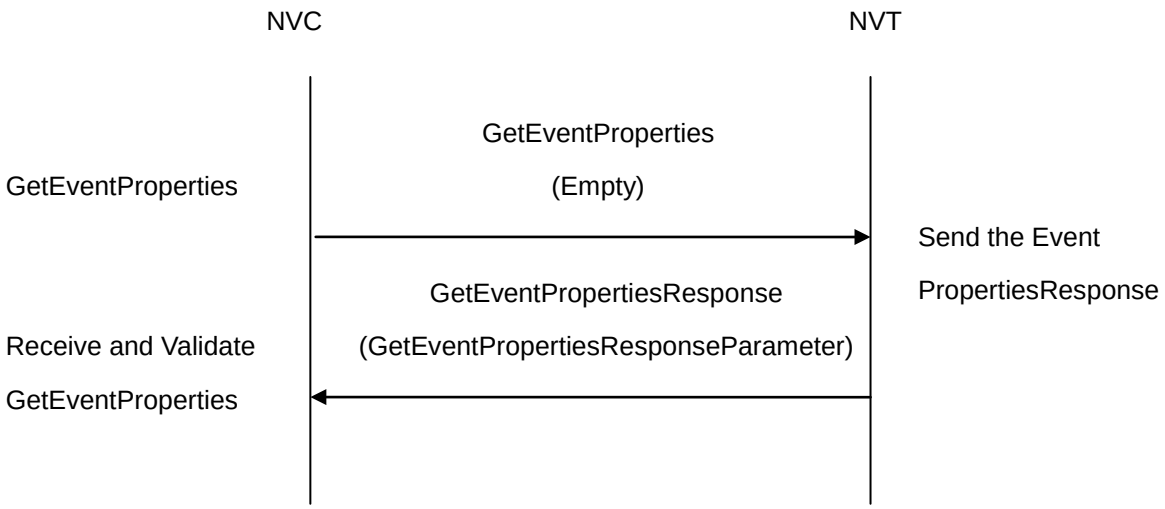
Requirement Level: MUST

Test Purpose: To verify NVT GetEventProperties command

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.

3. NVC will invoke GetEventProperties message to retrieve information about the FilterDialects, schema files and supported topics of the NVT.
4. Verify that NVT sends GetEventPropertiesResponse message
5. Validate that the mandatory TopicExpressionDialects are supported by the NVT (http://docs.oasis-open.org/wsn/t-1/TopicExpression/Concrete and http://www.onvif.org/ver10/tev/topicExpression/ConcreteSet)
6. Validate that the mandatory MessageContentFilterDialects is supported by the NVT (http://www.onvif.org/ver10/tev/messageContentFilter/ItemFilter)
7. Verify that the NVT returns a valid topic namespace
8. Verify that the DUT supports at least one TopicSet, validate that the TopicSet is well formed

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse message.

The DUT did not support the mandatory TopicExpressionDialects

The DUT did not support the mandatory MessageContentFilterDialects

The DUT did not support a valid topic namespace

The DUT did not support at least one TopicSet or the TopicSet is invalid

9.2 Basic Notification Interface

9.2.1 NVT BASIC NOTIFICATION INTERFACE - SUBSCRIBE

Test Label: Event handling SUBSCRIBE

ONVIF Core Specification Coverage: 12.1 (Basic Notification Interface)

Device Type: NVT

Command Under Test: Subscribe

WSDL Reference: event.wsdl

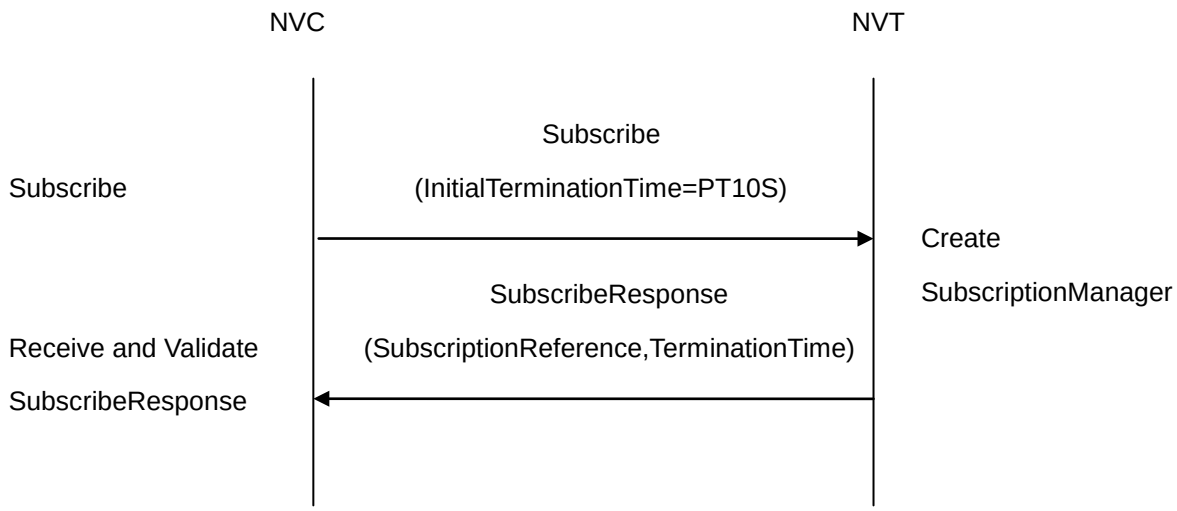
Requirement Level: MUST

Test Purpose: To verify NVT Subscribe command

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke Subscribe message to instantiate a Subscription Manager. The Subscribe message does not contain a TopicExpression or a Message Content Filter. The Message contains an InitialTerminationTime of 10s to ensure that the Subscription is terminated after end of this test.
4. Verify that NVT sends SubscribeResponse message. Validate that a valid SubscriptionReference is returned (valid EndpointReference); verify that valid values for CurrentTime and TerminationTime are returned (TerminationTime >= CurrentTime + InitialTerminationTime).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SubscribeResponse message.

The DUT did not return a valid SubscriptionReference

The DUT did not return valid values for CurrentTime and TerminationTime.

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

If NVT can not accept the set value to an InitialTerminationTime, NVC retries to send the Subscribe request with MinimumTime value which is contained in UnacceptableInitialTerminationTimeFault.

9.2.2 NVT BASIC NOTIFICATION INTERFACE - INVALID MESSAGE CONTENT FILTER

Test Label: Event handling SUBSCRIBE INVALID MESSAGE CONTENT FILTER

ONVIF Core Specification Coverage: 12.1 (Basic Notification Interface)

Device Type: NVT

Command Under Test: Subscribe

WSDL Reference: event.wsdl

Requirement Level: SHOULD

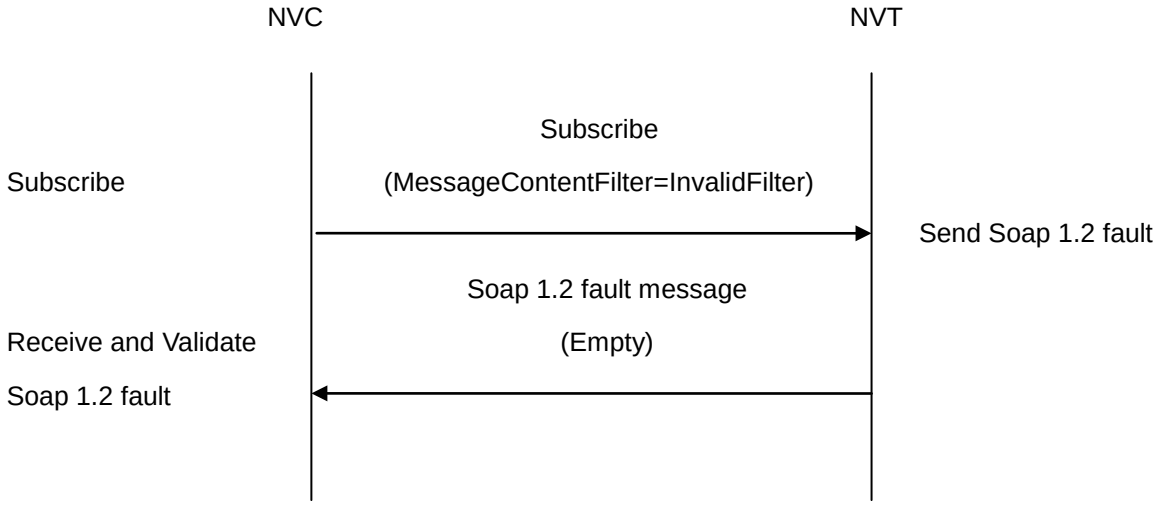
Test Purpose:

To verify that a correct error message "InvalidFilterFault" or “InvalidMessageContentExpressionFault” is returned if a Subscribe Request with an invalid MessageContentFilter is invoked.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.

3. NVC will invoke Subscribe message with an invalid Filter boolean((//tt:SimpleItem[@Name=”xyz” and @Value=”xyz”]).
4. Verify that the NVT generates an “InvalidFilterFault” or an “InvalidMessageContentExpressionFault” fault message.
5. Validate the fault message (valid utc time, valid description)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send an “InvalidFilterFault” or “InvalidMessageContentExpressionFault” fault message.

The DUT did not send a valid fault message

9.2.3 NVT BASIC NOTIFICATION INTERFACE - INVALID TOPIC EXPRESSION

Test Label: Event handling SUBSCRIBE INVALID FILTER-TOPIC EXPRESSION

ONVIF Core Specification Coverage: 12.1 (Basic Notification Interface)

Device Type: NVT

Command Under Test: Subscribe

WSDL Reference: event.wsdl

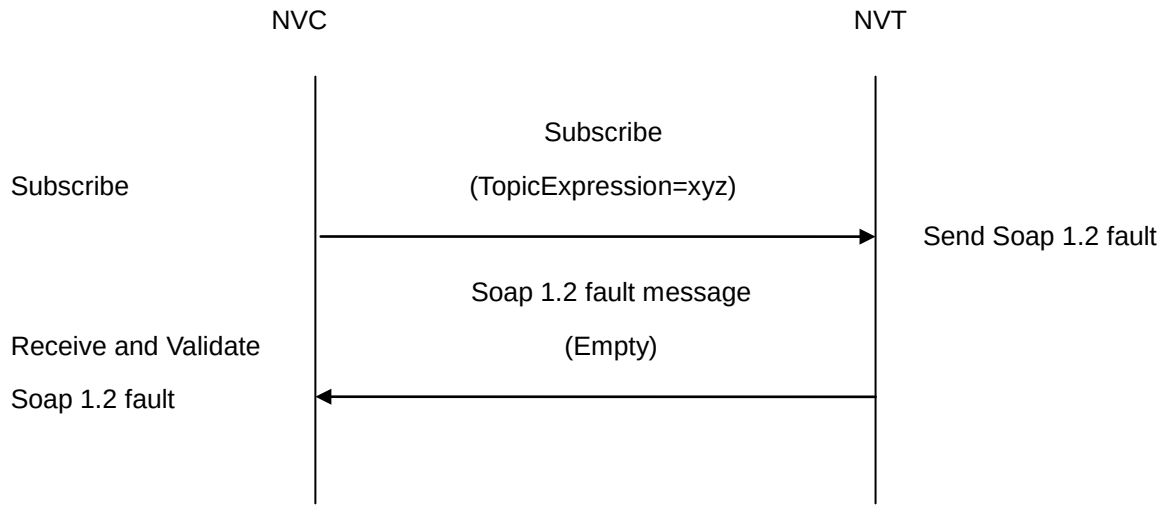
Requirement Level: SHOULD

Test Purpose: To verify that a correct error message "InvalidFilterFault" or “TopicNotSupported” is returned if a Subscribe Request with an invalid Topic Expression is invoked.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke Subscribe message with an invalid Topic Expression (boolean(///tt:SimpleItem[@Name=”xyz” and @Value=”xyz”]) .
4. Verify that the NVT generates an “InvalidFilterFault” or “TopicNotSupported” fault message.
5. Validate the fault message (valid utc time, valid description)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send an “InvalidFilterFault” or “TopicNotSupported” fault message.

The DUT did not send a valid fault message

9.2.4 NVT BASIC NOTIFICATION INTERFACE - RENEW

Test Label: Event handling Renew

ONVIF Core Specification Coverage: 12.1 (Basic Notification Interface)

Device Type: NVT

Command Under Test: Subscribe, Renew

Driving IP-based physical security through global standardization

WSDL Reference: event.wsdl

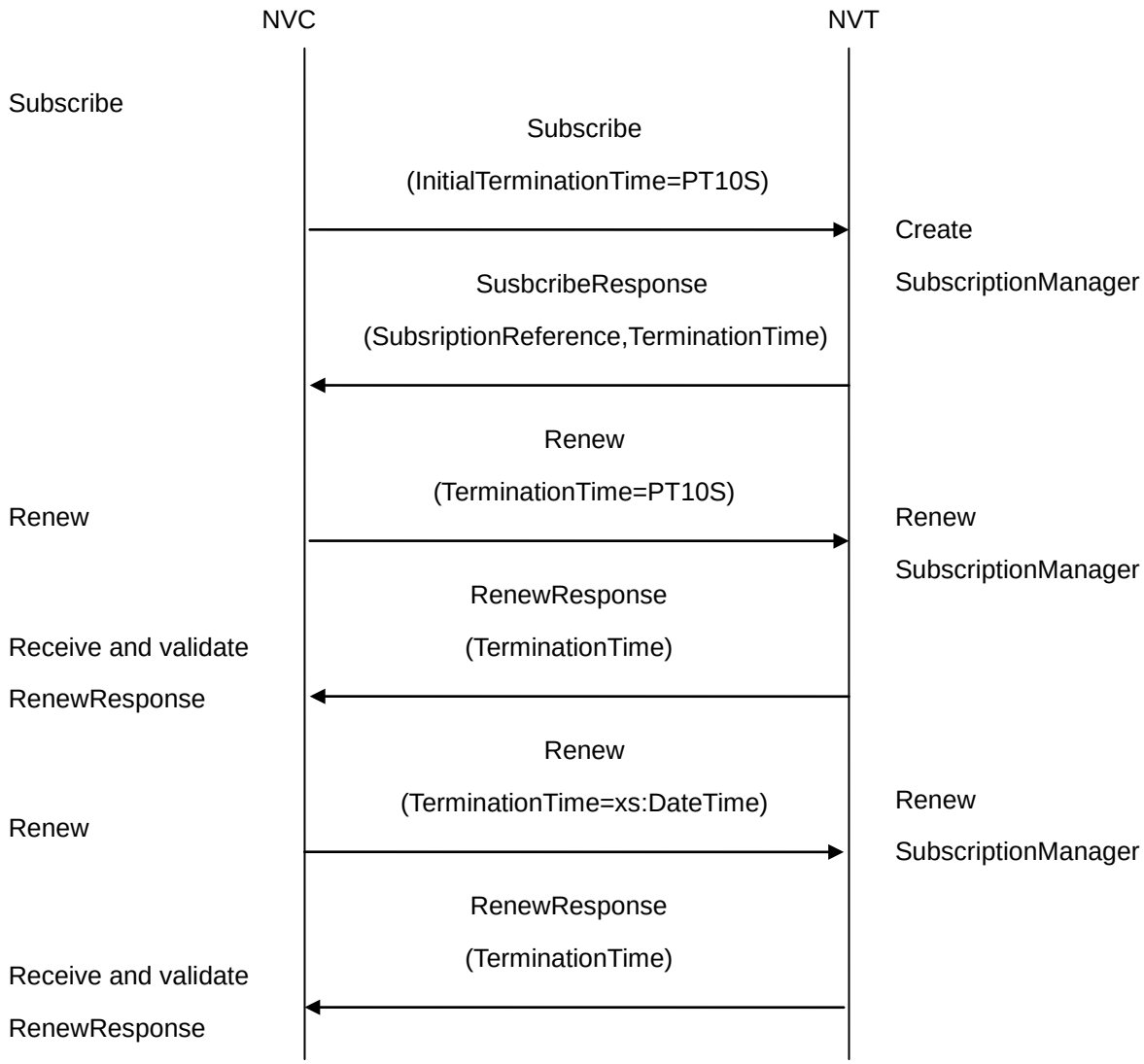
Requirement Level: MUST

Test Purpose: To verify Renew command

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke Subscribe message with an InitialTerminationTime of 10s
4. Verify that the DUT sends a SubscribeResponse.
5. Validate CurrentTime and TerminationTime (TerminationTime>=CurrentTime+ InitialTerminationTime)
6. NVC will invoke Renew command with a TerminationTime of 10s to ensure that the Subscription times out after 10s.
7. Verify that the DUT sends a RenewResponse
8. Verify CurrentTime and TerminationTime (TerminationTime>=CurrentTime+ TerminationTime)
9. NVC will invoke Renew command with a TerminationTime in xs:dateTime format. The TerminationTime shall be current time+ 10s.
10. Verify that the DUT sends a RenewResponse
11. Verify CurrentTime and TerminationTime (TerminationTime>=CurrentTime+ TerminationTime)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SubscribeResponse message.

The DUT did not send valid values for CurrentTime and TerminationTime.

The DUT did not send a RenewResponses

The DUT did not send valid values for CurrentTime and TerminationTime

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

If NVT can not accept the set value to an InitialTerminationTime, NVC retries to send the Subscribe request with MinimumTime value which is contained in UnacceptableInitialTerminationTimeFault.

If NVT can not accept the set value to a TerminationTime, NVC retries to send the Renew request MinimumTime value which is contained in UnacceptableTerminationTimeFault.

9.2.5 NVT BASIC NOTIFICATION INTERFACE - UNSUBSCRIBE

Test Label: Event handling UNSUBSCRIBE

Driving IP-based physical security through global standardization

ONVIF Core Specification Coverage: 12.1 (Basic Notification Interface)

Device Type: NVT

Command Under Test: Subscribe, Unsubscribe

WSDL Reference: event.wsdl

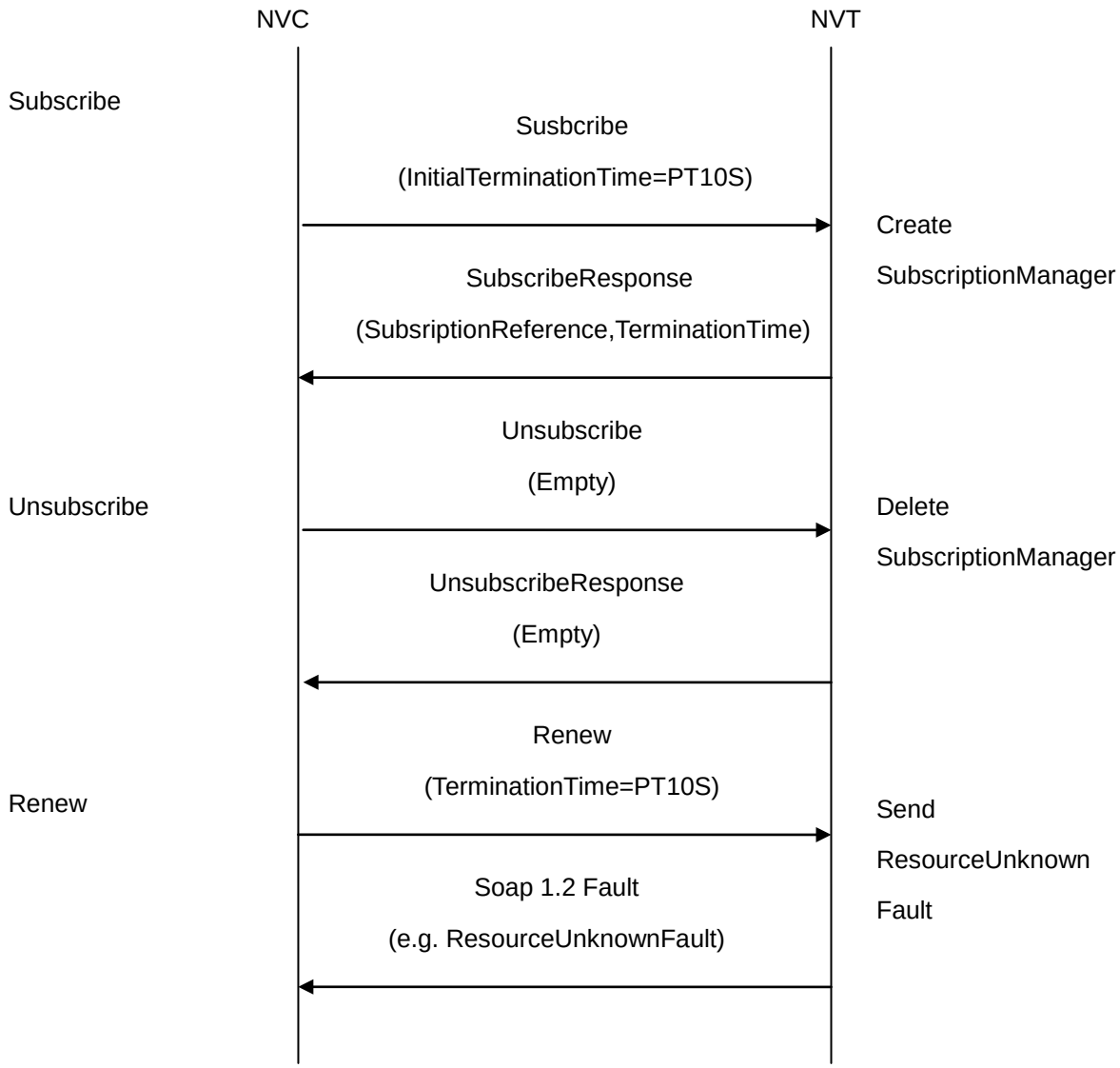
Requirement Level: MUST

Test Purpose: To verify Unsubscribe command

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke Subscribe message (InitialTerminationTime=PT10S) to instantiate an SubscriptionManager
4. Verify that the DUT sends a SubscribeResponse with valid values for SubscriptionManager, CurrentTime and TerminationTime.
5. NVC will invoke Unsubscribe command
6. Verify that the DUT sends a UnsubscribeResponse
7. NVC will invoke a Renew command to verify that the SubscriptionManager is deleted
8. Verify that the DUT sends a Soap 1.2 Fault (e.g. a “ResourceUnknown” fault message).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SubscribeResponse message.

The DUT did not send valid values for CurrentTime and TerminationTime (TerminationTime>=CurrentTime+InitialTerminationTime)

The DUT did not send an UnsubscribeResponse

The DUT did not send a Soap 1.2 fault message

Note: If NVT can not accept the set value to an InitialTerminationTime, NVC retries to send the Subscribe request with MinimumTime value which is contained in UnacceptableInitialTerminationTimeFault.

9.2.6 NVT BASIC NOTIFICATION INTERFACE - RESOURCE UNKNOWN

Test Label: Event handling RESOURCE UNKNOWN

ONVIF Core Specification Coverage: 12.1 (Basic Notification Interface)

Device Type: NVT

Command Under Test: Subscribe, Unsubscribe

WSDL Reference: event.wsdl

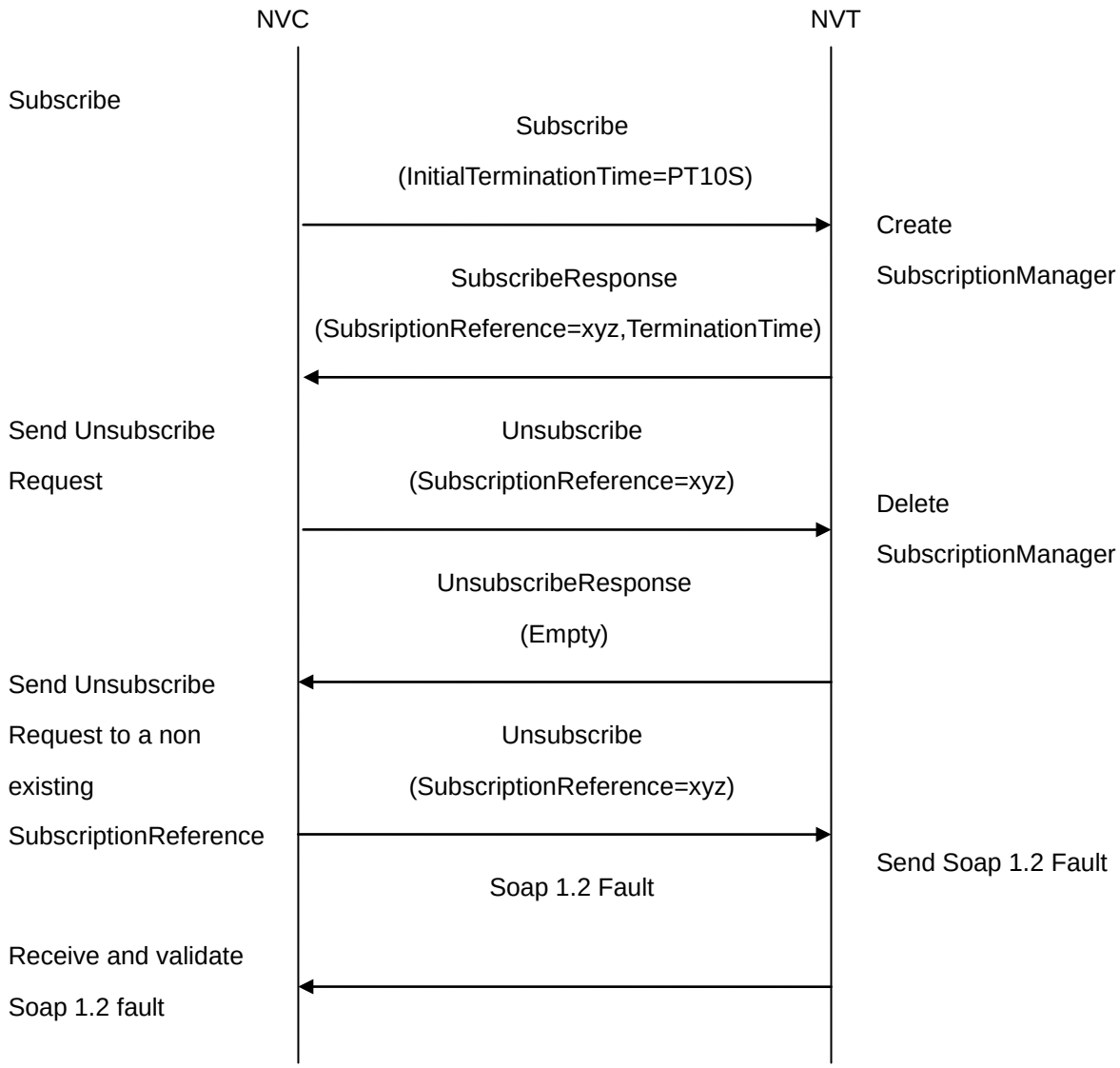
Requirement Level: SHOULD

Test Purpose: To verify that a Soap 1.2 Fault Message is returned if an UnsubscribeRequest to a non existing Subscription is sent

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke Subscribe message with a InitialTerminationTime of 10 s to ensure that the SubscriptionManager will be deleted after testing
4. Verify that the DUT sends a SubscribeResponse with valid values for SubscriptionReference and TerminationTime).

5. NVC will invoke Unsubscribe command to delete the SubscriptionManager
6. Verify that the DUT sends a UnsubscribeResponse
7. Send a second Unsubscribe Request to the just deleted SubscriptionReference
8. Verify that the DUT sends a Soap 1.2 Fault (e.g. a “ResourceUnknown” or a “UnableToDestroySubscription” Fault)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SubscribeResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send an UnsubscribeResponse

The DUT did not delete the SubscriptionManager

The DUT did not send a Soap 1.2 Fault

Note: If NVT can not accept the set value to an InitialTerminationTime, NVC retries to send the Subscribe request with MinimumTime value which is contained in UnacceptableInitialTerminationTimeFault.

9.2.7 NVT BASIC NOTIFICATION INTERFACE - NOTIFY

Test Label: Event handling NOTIFY

ONVIF Core Specification Coverage: 12.1 (Basic Notification Interface), 12.6 (SetSynchronizationPoint)

Device Type: NVT

Command Under Test: Subscribe, Unsubscribe, SetSynchronizationPoint, Notify

WSDL Reference: event.wsdl

Requirement Level: MUST

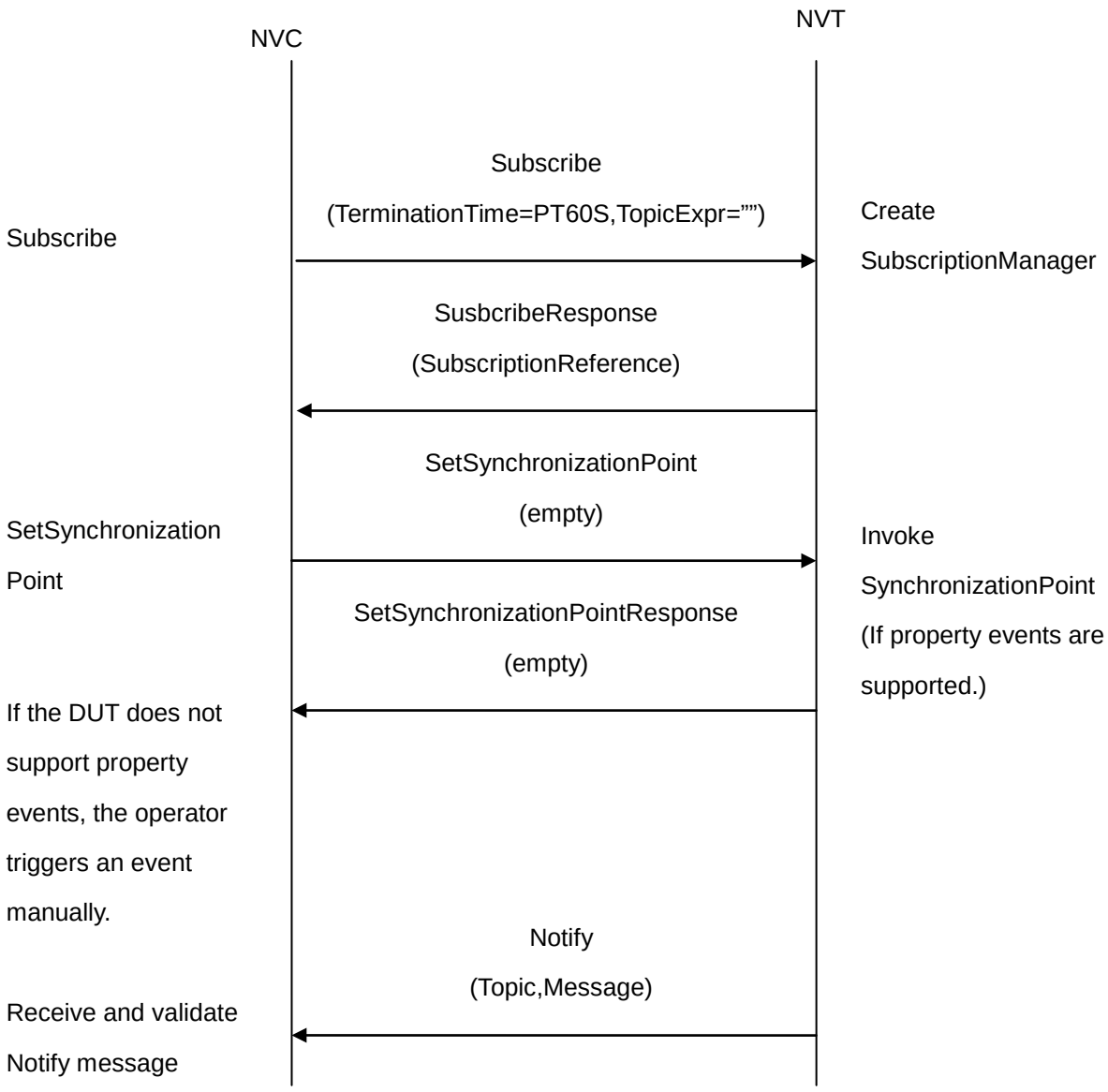
Test Purpose: To verify Notify message

Pre-Requisite: The DUT MUST provide at least one event.

The test operator has to ensure that the event is triggered and sent out. NVC will invoke a SetSynchronizationPoint request. If the DUT does not support property events of if it is not possible to invoke a SetSynchronizationPoint, the test operator has to trigger the event manually.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke Subscribe with an InitialTerminationTime of 60s to ensure that the SubscriptionManager is deleted after one minute
4. Verify that the DUT sends a SubscribeResponse with valid values for SubscriptionReference, TerminationTime and CurrentTime TerminationTime>=CurrentTime+InitialTerminationTime
5. Invoke SetSynchronizationPoint command to trigger an property event

6. Verify that DUT sends SetSynchronizationPointResponse
7. If the DUT does not support property events, the test operator has to trigger an event manually.
8. Verify that DUT sends Notify message(s)
9. Verify received Notify messages (correct value for Utc time, TopicExpression and wsnt:Message)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SubscribeResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a SetSynchronizationPointResponse

The DUT did not a Notify message

The DUT sends an invalid NOTIFY message

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

See Annex A.18 on how to compose Subscribe when the client is interested in receiving all event supported by the DUT.

9.2.8 NVT BASIC NOTIFICATION INTERFACE - NOTIFY FILTER

Test Label: Event handling NOTIFY FILTER

ONVIF Core Specification Coverage: 12.1 (Basic Notification Interface), 12.6 (SetSynchronizationPoint)

Device Type: NVT

Command Under Test: GetEventProperties, Subscribe, Unsubscribe, SetSynchronizationPoint, Notify

WSDL Reference: event.wsdl

Requirement Level: MUST

Test Purpose:

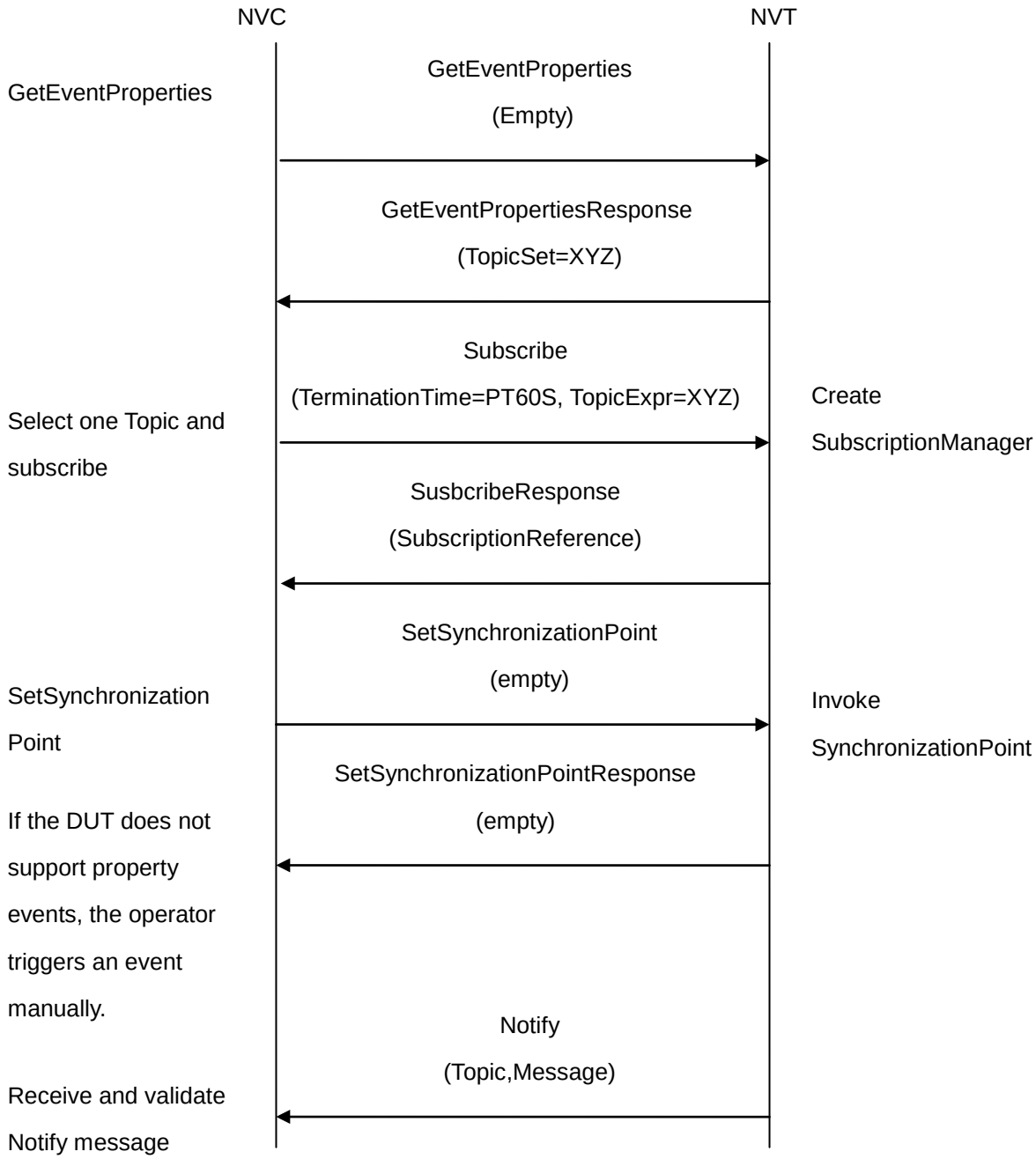
To verify that the device sends Notification messages; to verify if the DUT handles event filtering in a correct way.

Pre-Requisite: The DUT MUST provide at least one event.

The test operator has to ensure that the event is triggered and sent out. NVC will invoke a SetSynchronizationPoint request. If the DUT doesn't support property event or if it is not possible to invoke a SetSynchronizationPoint, the test operator has to trigger the event manually.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetEventProperties command
4. Verify that the DUT sends a GetEventPropertiesResponse, select one Topic.
5. NVC will invoke Subscribe with this Topic as Filter and a InitialTerminationTime of 60s to ensure that the SubscriptionManager is deleted after one minute
6. Verify that the DUT sends a SubscribeResponse with valid values for SubscriptionReference, TerminationTime and CurrentTime TerminationTime>=CurrentTime+60S
7. Invoke SetSynchronizationPoint command to trigger an event
8. Verify that DUT sends SetSynchronizationPointResponse
9. If the DUT does not support property events, the operator has to trigger an event manually.
10. Verify that DUT sends Notify message(s)
11. Check that at least one Notify message that contains a property event is returned. Verify this Notify messages (correct value for Utc time, TopicExpression and wsnt:Message).
12. Check if Notify message(s) are filtered according to the selected Filter.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send SubscribeResponse message.

The DUT did not send valid SubscriptionReference.

The DUT did not send a SetSynchronizationPointResponse

The DUT did not a Notify message that contains a property event

The DUT send an invalid NOTIFY message

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

9.3 Real-Time Pull-Point Notification Interface

9.3.1 NVT REALTIME PULLPOINT SUBSCRIPTION - CREATE PULL POINT SUBSCRIPTION

Test Label: event handling CREATE PULL POINT SUBSCRIPTION

ONVIF Core Specification Coverage: 12.2.1 (CreatePullPointSubscription)

Device Type: NVT

Command Under Test: CreatePullPointSubscription

WSDL Reference: event.wsdl

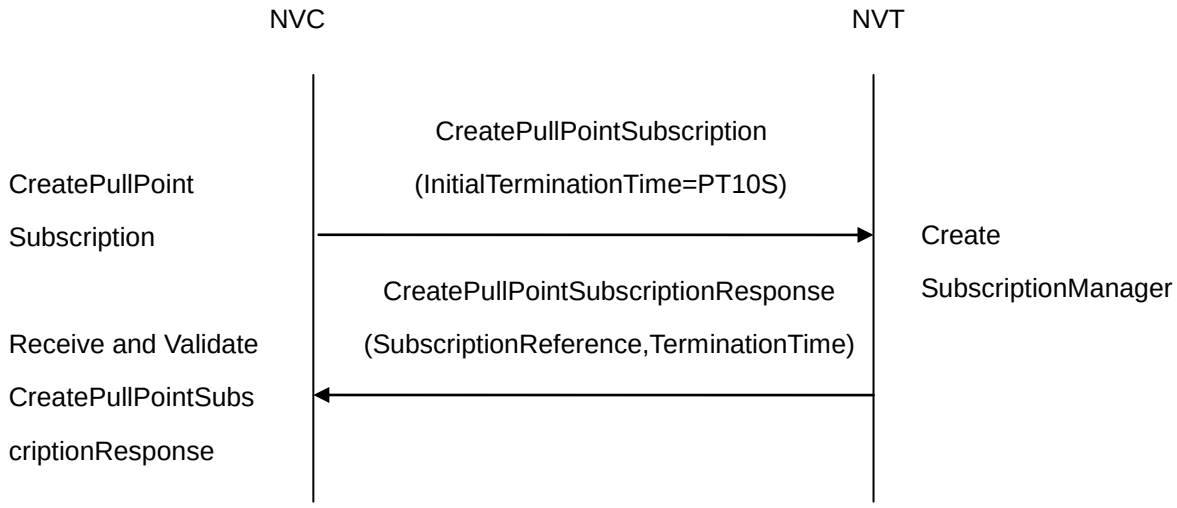
Requirement Level: MUST

Test Purpose: To verify NVT CreatePullPointSubscription command

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke CreatePullPointSubscription message to instantiate a Subscription Manager. The CreatePullPointSubscription message does not contain a TopicExpression or Message Content Filter. The Message contains an InitialTerminationTime of 10s to ensure that the SubscriptionManager is terminated after end of this test.

4. Verify that NVT sends CreatePullPointSubscriptionResponse message
5. Validate that valid values for SubscriptionReference CurrentTime and TerminationTime are returned (TerminationTime>=CurrentTime+InitialTerminationTime)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not return a valid SubscriptionReference

The DUT did not return valid values for CurrentTime and TerminationTime.

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

If NVT can not accept the set value to an InitialTerminationTime, NVC retries to send the CreatePullpointSubscription request with MinimumTime value which is contained in UnacceptableInitialTerminationTime fault.

9.3.2 NVT REALTIME PULLPOINT SUBSCRIPTION - INVALID MESSAGE CONTENT FILTER

Test Label: event handling CREATE PULL POINT SUBSCRIPTION – INVALID MESSAGE CONTENT FILTER

ONVIF Core Specification Coverage: 12.2.1 (CreatePullPointSubscription)

Device Type: NVT

Command Under Test: CreatePullPointSubscription

WSDL Reference: event.wsdl

Requirement Level: SHOULD

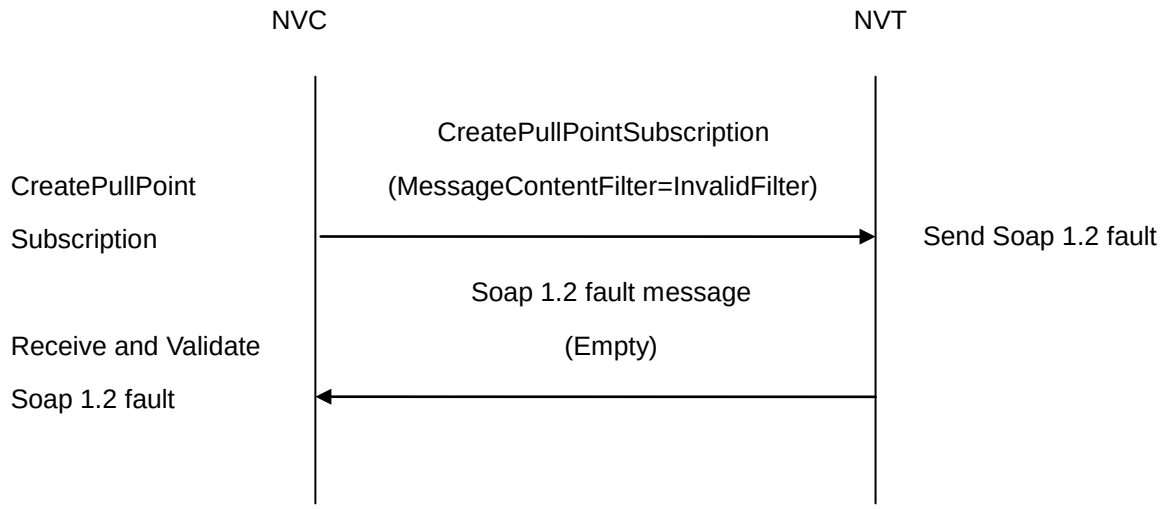
Test Purpose:

To verify that a correct error message "InvalidFilterFault" or “InvalidMessageContentExpressionFault” is returned if a CreatePullPointSubscription command with an invalid MessageContentFilter is invoked.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke CreatePullPointSubscription message with an invalid Filter boolean((//tt:SimpleItem[@Name=”xyz” and @Value=”xyz”]).
4. Verify that the NVT generates an “InvalidFilterFault” or an “InvalidMessageContentExpressionFault” fault message. Validate that utc time and description are correct.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send an “InvalidFilterFault” or “InvalidMessageContentExpressionFault” fault message.

The DUT did not send a valid fault message

9.3.3 NVT REALTIME PULLPOINT SUBSCRIPTION - INVALID TOPIC EXPRESSION

Test Label: Event handling REALTIME PULLPOINT INVALID FILTER-TOPIC EXPRESSION

ONVIF Core Specification Coverage: 12.2.1 (CreatePullPointSubscription)

Device Type: NVT

Command Under Test: CreatePullPointSubscription

WSDL Reference: event.wsdl

Requirement Level: SHOULD

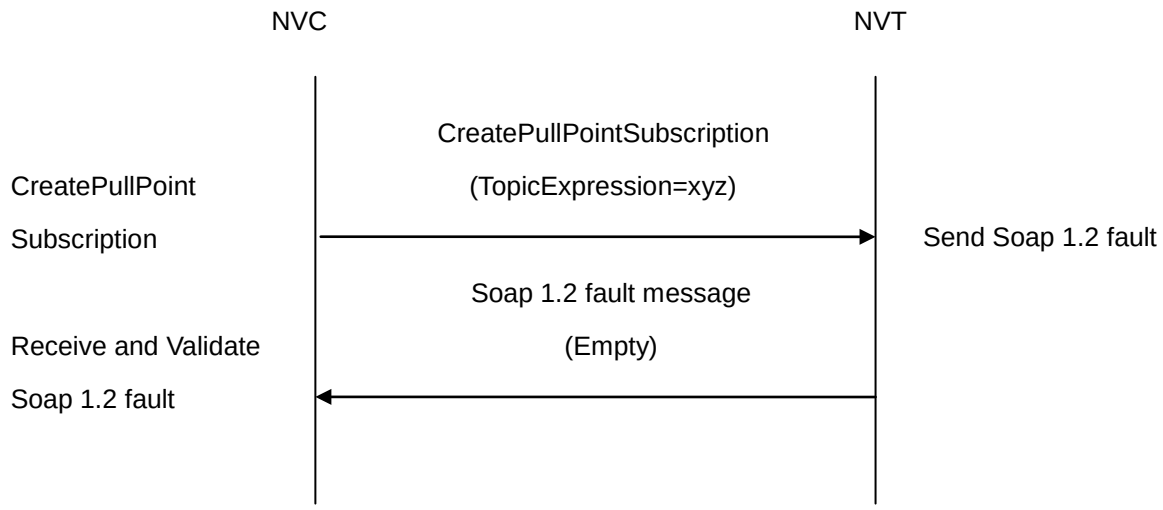
Test Purpose:

To verify that a correct error message "InvalidFilterFault" or “TopicNotSupported” is returned if a CreatePullPointSubscription command with an invalid TopicExpression is invoked.

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke CreatePullPointSubscription message with an invalid Topic Expression “xyz”.
4. Verify that the NVT generates an “InvalidFilterFault” or “TopicNotSupported” fault message.
5. Validate valid the fault message (valid utc time, valid description)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send an “InvalidFilterFault” or “TopicNotSupported” fault message.

The DUT did not send a valid fault message

9.3.4 NVT REALTIME PULLPOINT SUBSCRIPTION - RENEW

Test Label: Event handling RealtimePullPoint Renew

ONVIF Core Specification Coverage: 12.1 (Basic Notification Interface), 12.2.1 (CreatePullPointSubscription)

Device Type: NVT

Command Under Test: CreatePullPointSubscription, Renew

WSDL Reference: event.wsdl

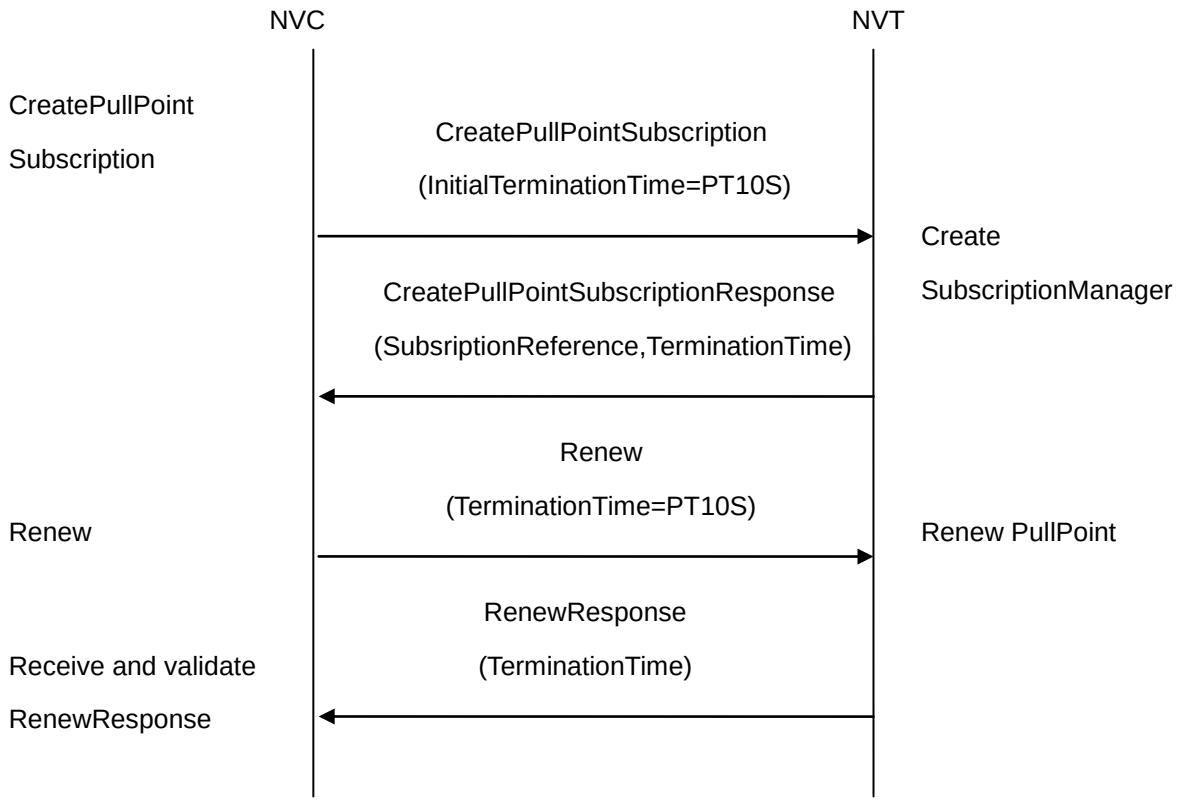
Requirement Level: MUST

Test Purpose: To verify Renew command

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke CreatePullPointSubscription message with an InitialTerminationTime of 10s
4. Verify that the DUT sends a CreatePullPointSubscriptionResponse.
5. Validate CurrentTime and TerminationTime (TerminationTime>=CurrentTime+10s)
6. NVC will invoke Renew command with a TerminationTime of 10s to ensure that the Subscription times out after the test
7. Verify that the DUT sends a RenewResponse
8. Verify CurrentTime and TerminationTime (TerminationTime>=CurrentTime+10s)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SubscribeResponse message.

The DUT did not send valid values for CurrentTime and TerminationTime.

The DUT did not send a RenewResponse

The DUT did not send valid values for CurrentTime and TerminationTime

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

If NVT can not accept the set value to an InitialTerminationTime, NVC retries to send the CreatePullpointSubscription request with MinimumTime value which is contained in UnacceptableInitialTerminationTime fault.

9.3.5 NVT REALTIME PULLPOINT SUBSCRIPTION - UNSUBSCRIBE

Test Label: Event handling RealTime PullPoint UNSUBSCRIBE

ONVIF Core Specification Coverage: 12.1 (Basic Notification Interface), 12.2.2 (CreatePullPointSubscription)

Device Type: NVT

Command Under Test: CreatePullPointSubscription, Unsubscribe, Renew

WSDL Reference: event.wsdl

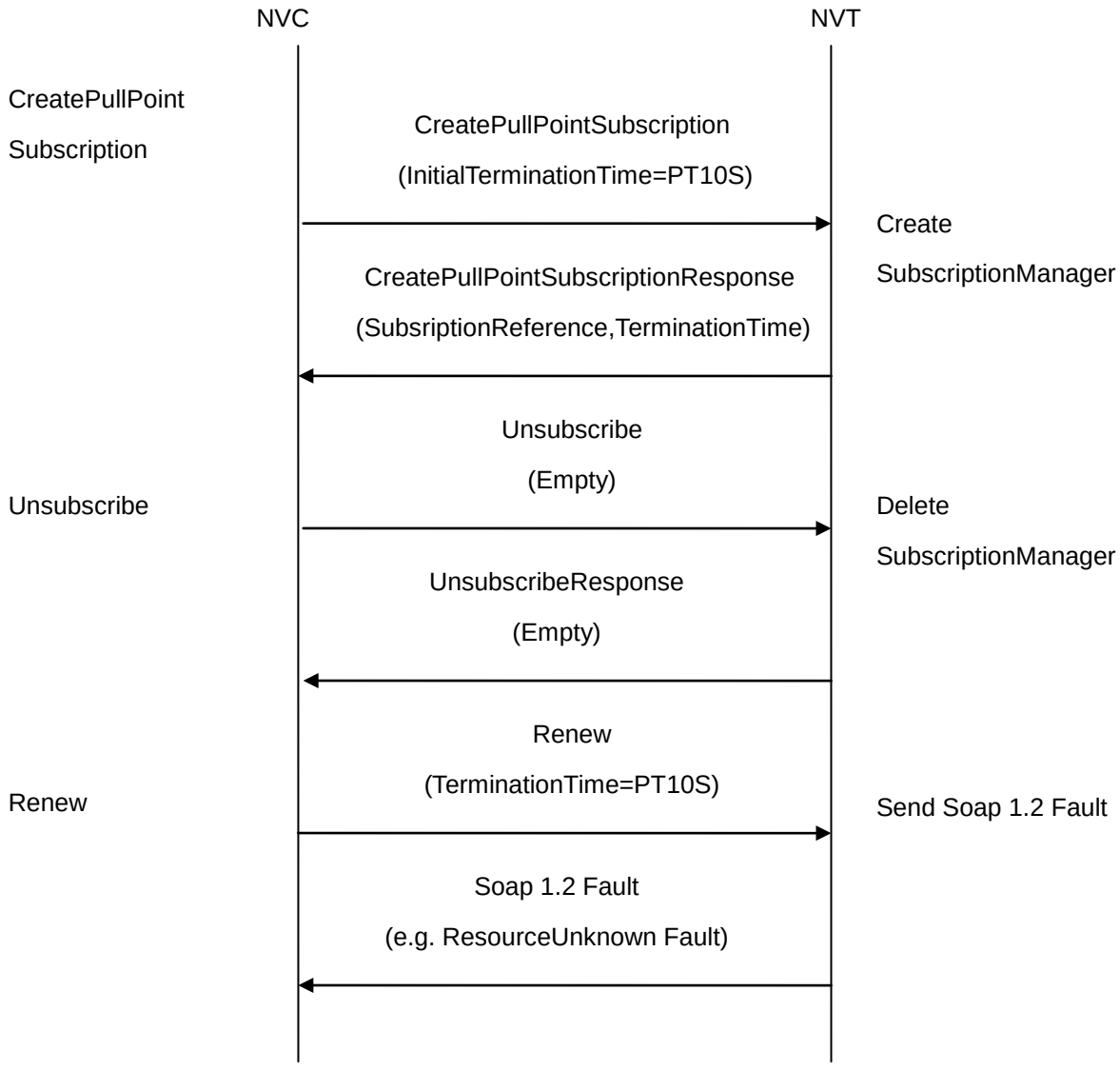
Requirement Level: MUST

Test Purpose: To verify Unsubscribe command

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke CreatePullPointSubscription message (InitialTerminationTime=PT10S) to instantiate an SubscriptionManager

4. Verify that the DUT sends a CreatePullPointSubscriptionResponse. Validate CurrentTime and TerminationTime
5. NVC will invoke Unsubscribe command to terminate the SubscriptionManager
6. Verify that the DUT sends a UnsubscribeResponse
7. NVC will invoke a Renew command to verify that the SubscriptionManager is deleted
8. Verify that the DUT sends a Soap 1.2 Fault (e.g. a “ResourceUnknown” fault message)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid values for CurrentTime and TerminationTime (TerminationTime=CurrentTime+10s)

The DUT did not send an UnsubscribeResponse

The DUT did not send a Soap 1.2.

Note: If NVT can not accept the set value to an InitialTerminationTime, NVC retries to send the CreatePullpointSubscription request with MinimumTime value which is contained in UnacceptableInitialTerminationTime fault.

9.3.6 NVT REALTIME PULLPOINT SUBSCRIPTION - TIMEOUT

Test Label: Event handling Realtime Pull Point Timeout

ONVIF Core Specification Coverage: 12.2.2 (CreatePullPointSubscription), 12.1 (Basic Notification Interface)

Device Type: NVT

Command Under Test: CreatePullPointSubscription

WSDL Reference: event.wsdl

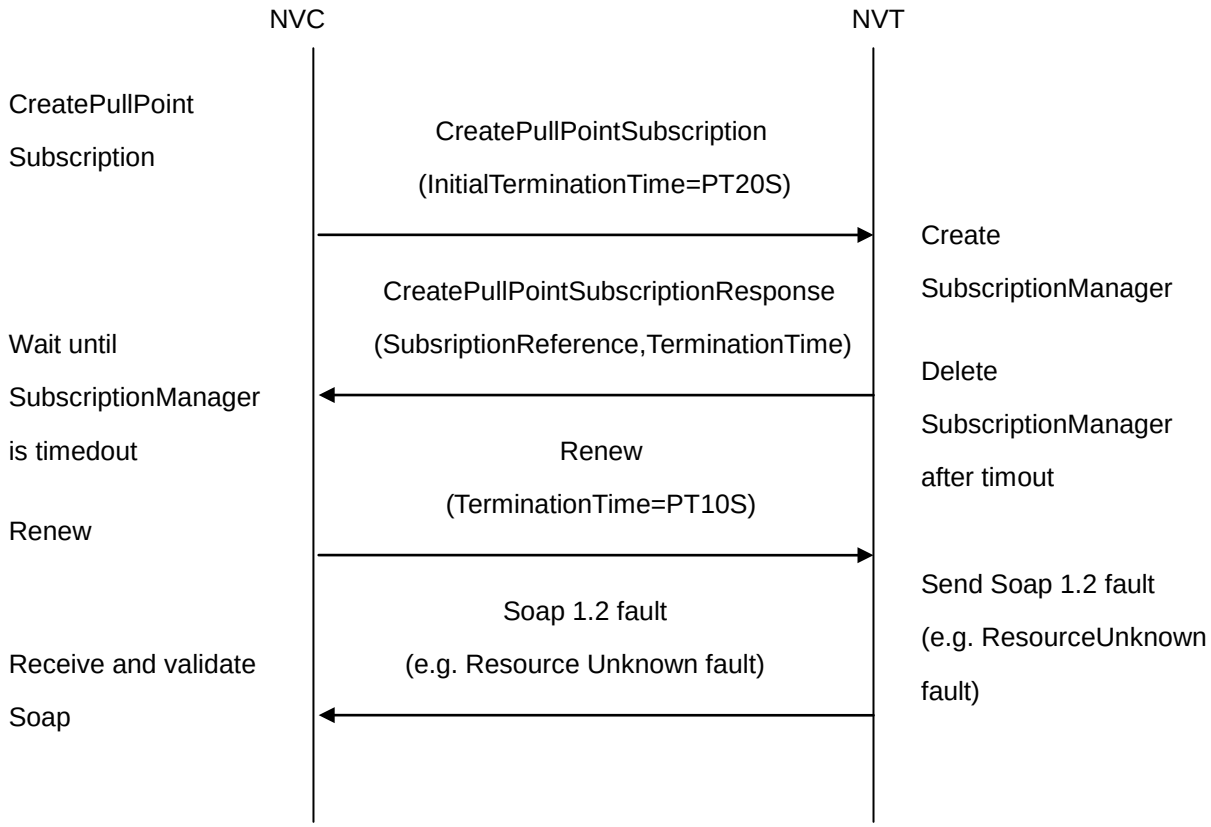
Requirement Level: MUST

Test Purpose: To verify if a SubscriptionManager times out correctly

Pre-Requisite: None

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke CreatePullPointSubscription message with a suggested timeout of PT20S.
4. Verify that the DUT sends a CreatePullPointSubscriptionResponse.
5. Validate CurrentTime and TerminationTime and SubscriptionReference
6. Wait 20 s (SubscriptionManager timeout)
7. NVC will invoke Renew command to check if the SubscriptionManager is timed out.
8. Verify that the DUT sends a Soap 1.2 fault (e.g. a “ResourceUnknown” fault)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid values for CurrentTime and TerminationTime (TerminationTime >= CurrentTime+10s)

The DUT did not send a Soap 1.2 fault

Note: If NVT can not accept the set value to an InitialTerminationTime, NVC retries to send the CreatePullpointSubscription request with MinimumTime value which is contained in UnacceptableInitialTerminationTime fault.

9.3.7 NVT REALTIME PULLPOINT SUBSCRIPTION - PULLMESSAGES

Test Label: event handling NVT REALTIME PULL POINT INTERFACE PullMessages

ONVIF Core Specification Coverage: 12.2.1(CreatePullPointSubscription), 12.6 (SetSynchronizationPoint), 12.2.2 (PullMessages)

Device Type: NVT

Command Under Test: CreatePullPointSubscription, SetSynchronizationPoint, PullMessages

WSDL Reference: event.wsdl

Requirement Level: MUST

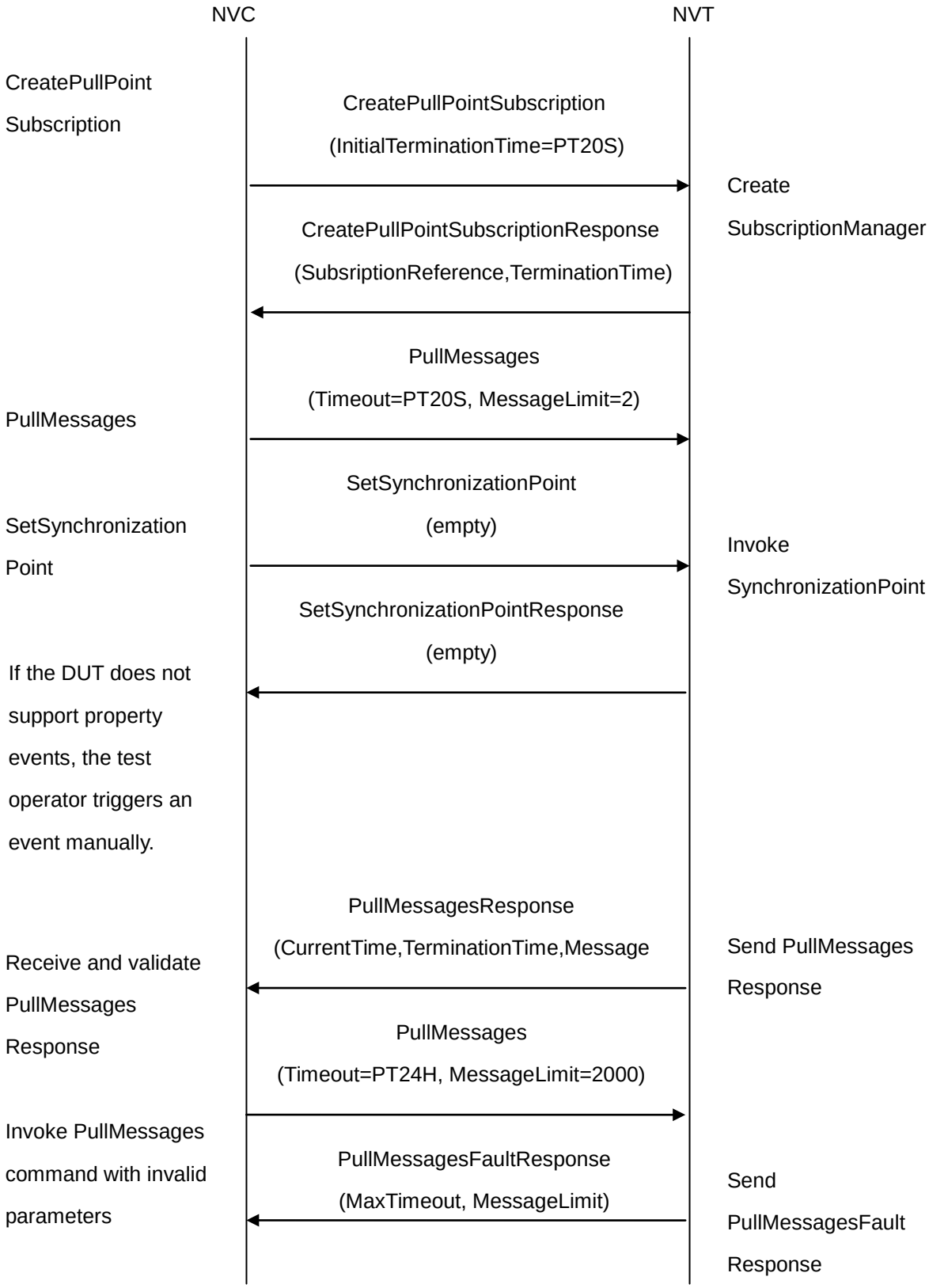
Test Purpose: To verify PullMessages command

Pre-Requisite: The DUT MUST provide at least one event.

The test operator has to ensure that the event is triggered and sent out. NVC will invoke a SetSynchronizationPoint request. If the device does not support property events or if it is not possible to invoke a SetSynchronizationPoint, the test operator has to trigger event manually.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
CreatePullPoint Subscription
CreatePullPointSubscription
(InitialTerminationTime=PT20S)
Create SubscriptionManager
CreatePullPointSubscriptionResponse
(SubsriptionReference,TerminationTime)
PullMessages
(Timeout=PT20S, MessageLimit=2)
PullMessages
SetSynchronizationPoint
(empty)
SetSynchronization Point
Invoke SynchronizationPoint
SetSynchronizationPointResponse
(empty)
If the DUT does not support property events, the test operator triggers an event manually.
PullMessagesResponse
(CurrentTime,TerminationTime,Message
Send PullMessages Response
Receive and validate PullMessages Response
PullMessages
(Timeout=PT24H, MessageLimit=2000)
Invoke PullMessages command with invalid parameters
PullMessagesFaultResponse
(MaxTimeout, MessageLimit)
Send PullMessagesFault Response

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke CreatePullPointSubscription message with a suggested timeout of PT20S.
4. Verify that the DUT sends a CreatePullPointSubscriptionResponse. Validate that correct values for CurrentTime and TerminationTime and SubscriptionReference are returned
5. NVC will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 2
6. NVC will invoke SetSynchronizationPoint command to trigger an event
7. Verify that the DUT sends a SetSynchronizationPointResponse
8. If the DUT does not support property events, the operator has to trigger an event manually.
9. Verify that the DUT sends a PullMessagesResponse that contains at least one NotificationMessage that represents a property.
10. Verify NotificationMessage (a maximum number of 2 Notification Messages is included in the PullMessages Response; well formed and valid values for CurrentTime and TerminationTime (TerminationTime>CurrentTime)
11. NVC will invoke PullMessages request with invalid parameters (Timeout=24h; MessageLimit=2000)
12. Verify that the DUT sends a PullMessagesFaultResponse.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid values for CurrentTime and TerminationTime (TerminationTime > CurrentTime)

The DUT did not send a SetSynchronizationPointResponse

The DUT did not send a PullMessagesResponse

The PullMessagesResponse does not contain a NotificationMessage

The PullMessagesResponse contains more than 2 NotificationMessages

The NotificationMessages are not well formed

The PullMessagesResponse contains invalid values for Current or TerminationTime

The DUT did not send a PullMessagesFaultResponse.

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

It may be possible that some other events than the event which is being verified will be sent as PullMessages response during this test case. In such case, NVC should simply discard such response and they retry PullMessages request for the very event for verification.

During test case execution, it should be guaranteed that the DUT should not delete the property event which is being used for verification.

If NVT can not accept the set value to Timeout or MessageLimit, NVC retries to send the PullMessage message with Timeout and MessageLimit which is contained in PullMessagesFaultResponse.

See Annex A.18 on how to compose Subscribe when the client is interested in receiving all event supported by the DUT.

9.3.8 NVT REALTIME PULLPOINT SUBSCRIPTION - PULLMESSAGES FILTER

Test Label: event handling NVT REALTIME PULL POINT INTERFACE PullMessages Filter

ONVIF Core Specification Coverage: 12.2.1(CreatePullPointSubscription), 12.6 (SetSynchronizationPoint), 12.2.2 (PullMessages), 12.5.5(MessageFilter)

Device Type: NVT

Command Under Test: GetEventProperties, CreatePullPointSubscription, SetSynchronizationPoint, PullMessages

WSDL Reference: event.wsdl

Requirement Level: MUST

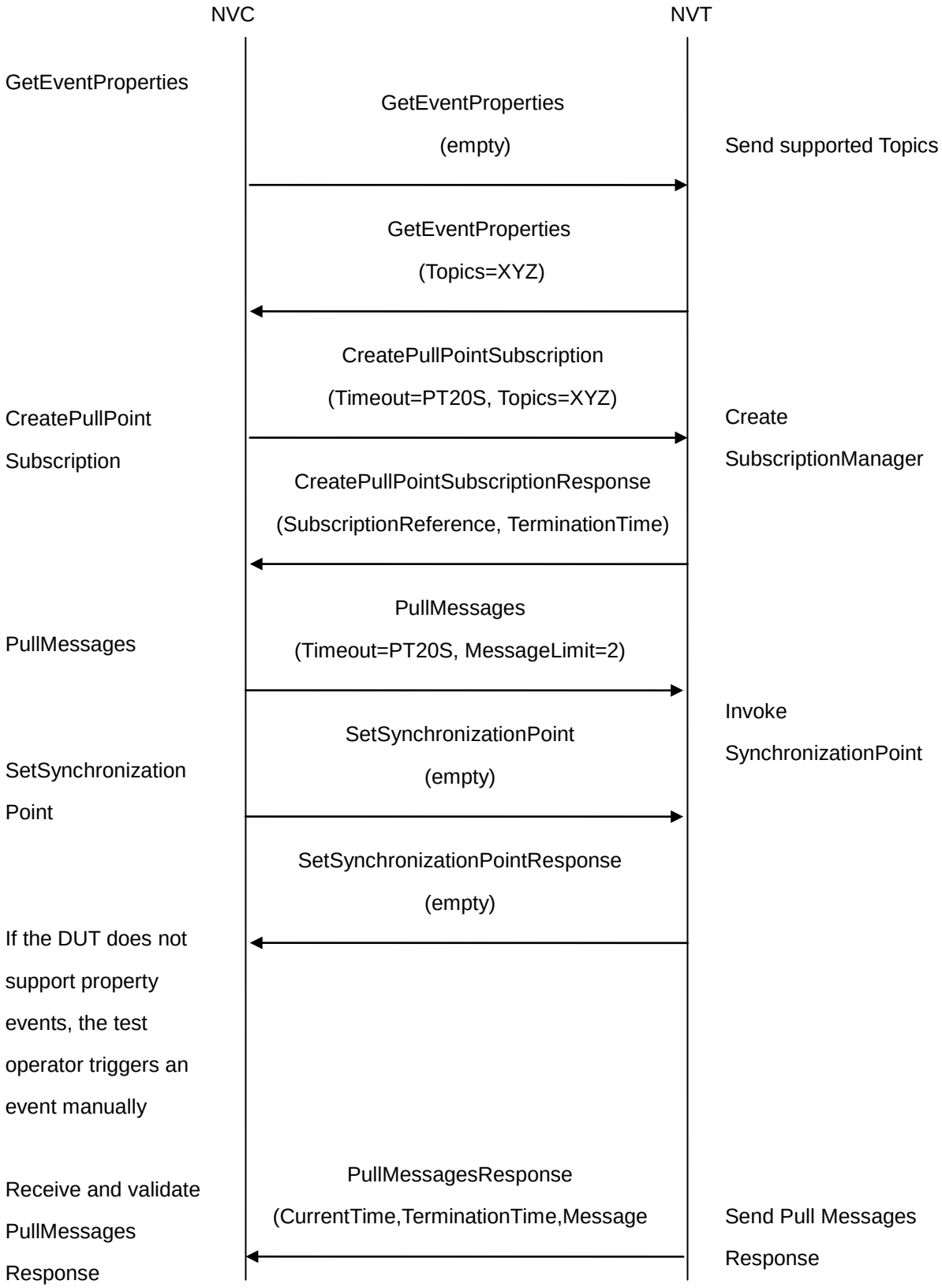
Test Purpose: To verify PullMessages command

Pre-Requisite: The DUT MUST provide at least one event.

The test operator has to ensure that the event is triggered and sent out. NVC will invoke a SetSynchronizationPoint request. If the DUT does not support property events of if it is not possible to invoke a SynchronizationPoint, the test operator has to trigger an event manually.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
GetEventProperties
GetEventProperties
(empty)
Send supported Topics
GetEventProperties
(Topics=XYZ)
CreatePullPointSubscription
(Timeout=PT20S, Topics=XYZ)
CreatePullPoint
Subscription
Create
SubscriptionManager
CreatePullPointSubscriptionResponse
(SubscriptionReference, TerminationTime)
PullMessages
(Timeout=PT20S, MessageLimit=2)
PullMessages
Invoke
SynchronizationPoint
SetSynchronizationPoint
(empty)
SetSynchronization
Point
SetSynchronizationPointResponse
(empty)
If the DUT does not support property events, the test operator triggers an event manually
PullMessagesResponse
(CurrentTime,TerminationTime,Message
Receive and validate PullMessages Response
Send Pull Messages
Response

Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC invokes GetEventProperties command to retrieve the supported Topics.
4. Verify that the DUT sends a GetEventPropertiesResponse; select one Topic
5. NVC will invoke CreatePullPointSubscription message with a suggested timeout of PT20S and a Filter including the selected Topic.
6. Verify that the DUT sends a CreatePullPointSubscriptionResponse.
7. Validate CurrentTime and TerminationTime and SubscriptionReference
8. NVC will invoke PullMessages command with a PullMessagesTimeout of 20s and a MessageLimit of 2
9. NVC will invoke SetSynchronizationPoint command to trigger an property event
10. Verify that the DUT sends a SetSynchronizationPointResponse
11. If the DUT does not support property events, the operator has to trigger an event manually.
12. Verify that the DUT sends a PullMessagesResponse that contains at least one NotificationMessage
13. Verify that that a maximum number of 2 Notification Messages is included in the PullMessages Response.
14. Verify that at least one property event is returned.
15. Verify that this NotificationMessage is well formed; Verify CurrentTime and TerminationTime (TerminationTime>CurrentTime)
16. Verify that the Topic of the NotificationMessage matches the filter

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send a GetEventPropertiesResponse

The DUT did not send a valid GetEventPropertiesResponse (containing at least one Topic)

The DUT did not send CreatePullPointSubscriptionResponse message.

The DUT did not send valid values for CurrentTime and TerminationTime (TerminationTime > CurrentTime)

The DUT did not send a SetSynchronizationPointResponse

The DUT did not send a PullMessagesResponse

The PullMessagesResponse does not contain a NotificationMessage

The PullMessagesResponse contains more than 2 NotificationMessages

The PullMessagesResponse does not contain at least one event

The NotificationMessages are not well formed

The NotificationMessage contains to a topic that was not requested

The PullMessagesResponse contains invalid values for Current or TerminationTime

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

It may be possible that some other events than the event which is being verified will be sent as PullMessages response during this test case. In such case, NVC should simply discard such response and they retry PullMessages request for the very event for verification.

During test case execution, it should be guaranteed that the DUT should not delete the property event which is being used for verification.

If NVT can not accept the set value to Timeout or MessageLimit, NVC retries to send the PullMessage message with Timeout and MessageLimit which is contained in PullMessagesFaultResponse.

9.4 Notification Streaming Interface

9.4.1 NVT NOTIFICATION STREAMING

Test Label: event handling Notification Streaming

ONVIF Core Specification Coverage: 10.2.1 (CreateProfile), 10.4.1 (GetVideoSourceConfigurations), 10.10.1 (GetMetadataConfigurations), 10.2.4 (AddVideoSourceConfiguration), 10.2.10 (AddMetadataConfiguration), 10.10.5 (SetMetadataConfiguration), 10.11.1 (GetStreamUri), 10.14.1 (SetSynchronizationPoint), 10.2.18 (DeleteProfile), 12.3 (Notification Streaming Interface)

Device Type: NVT

Command Under Test: CreateProfile, GetVideoSourceConfigurations, GetMetadataConfigurations, AddVideoSourceConfiguration, AddMetadataConfiguration, SetMetadataConfiguration, GetStreamUri, SetSynchronizationPoint, DeleteProfile

WSDL Reference: media.wsdl

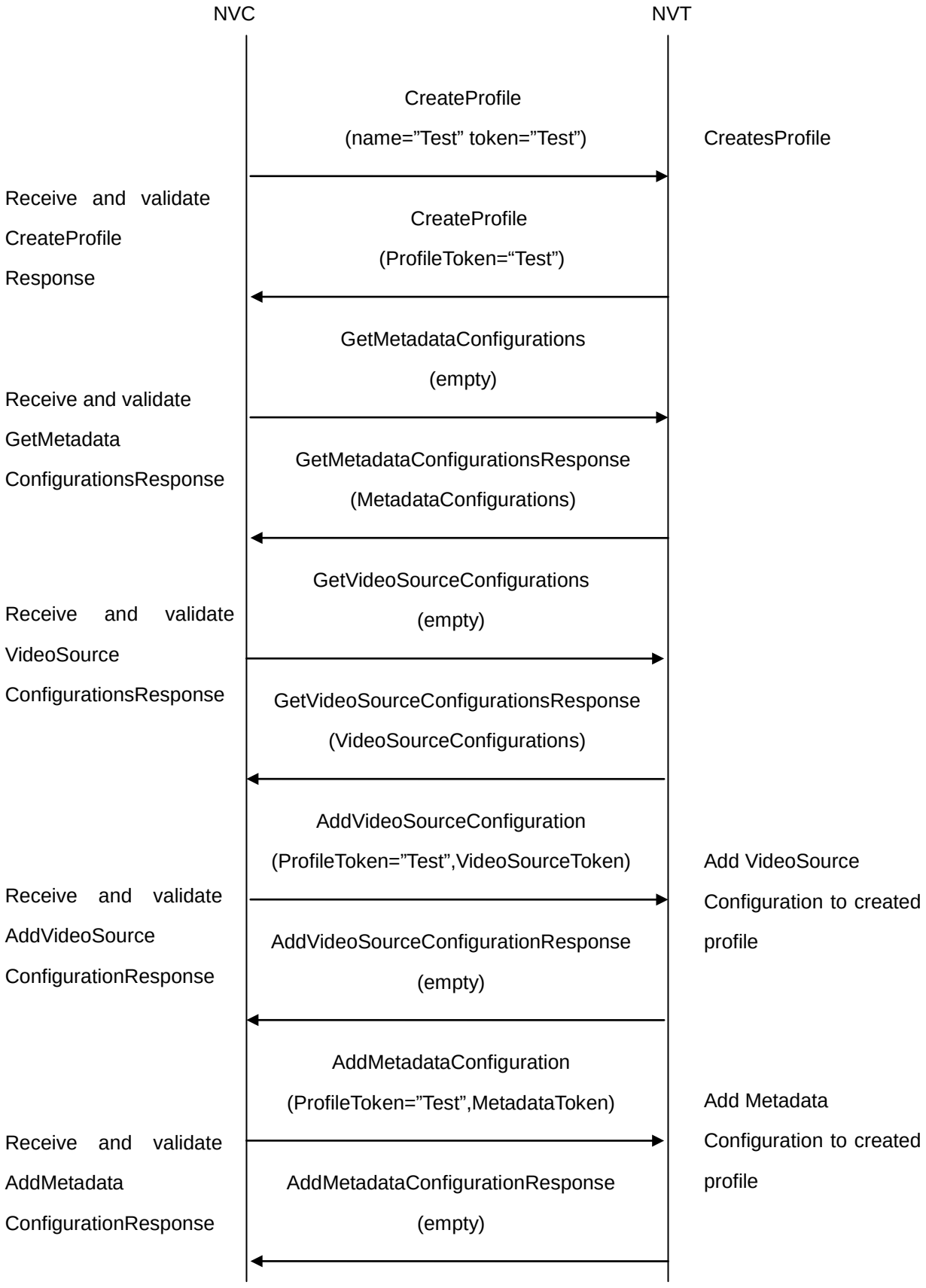
Requirement Level: MUST

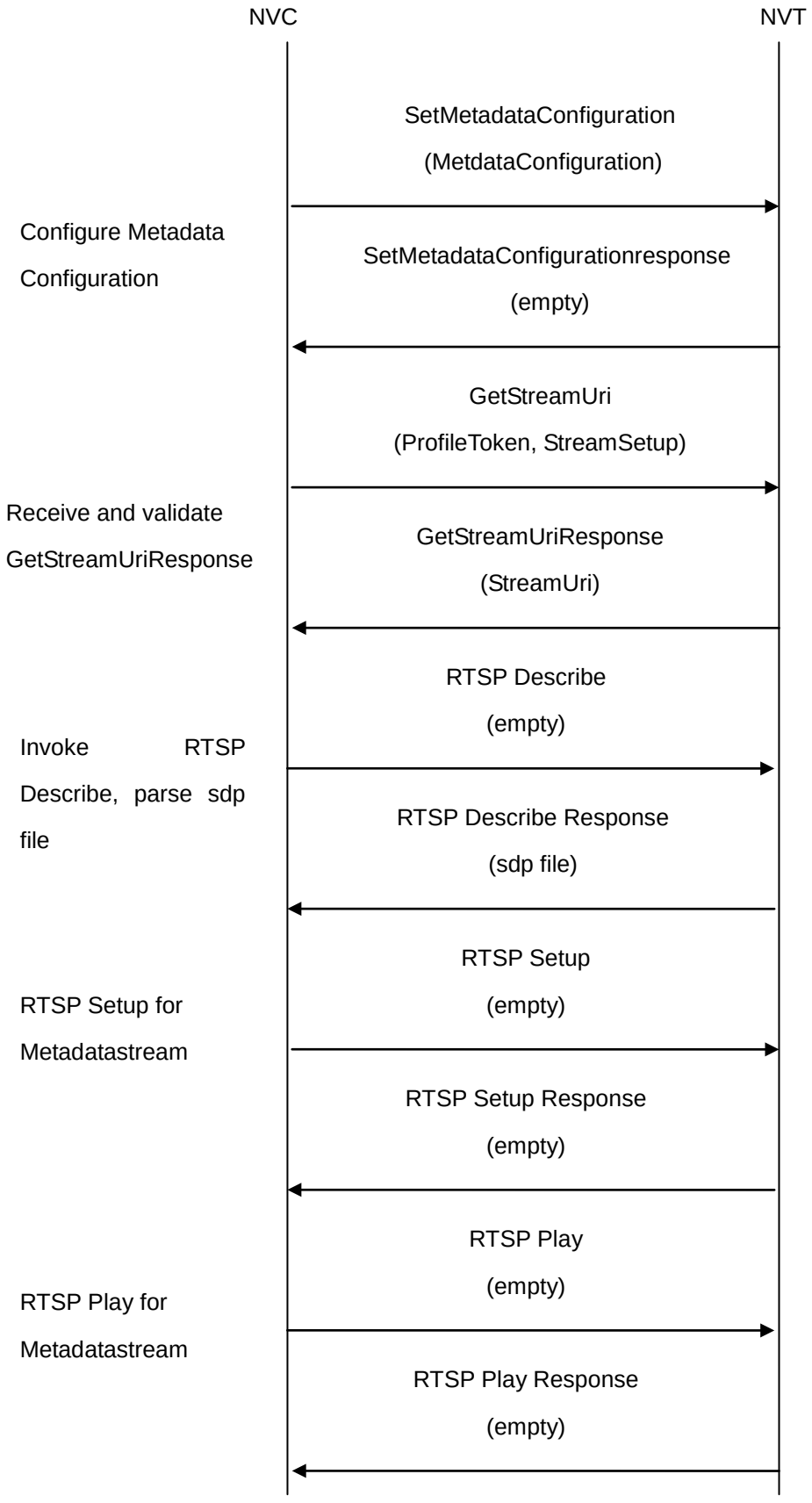
Test Purpose: To verify Notification Streaming

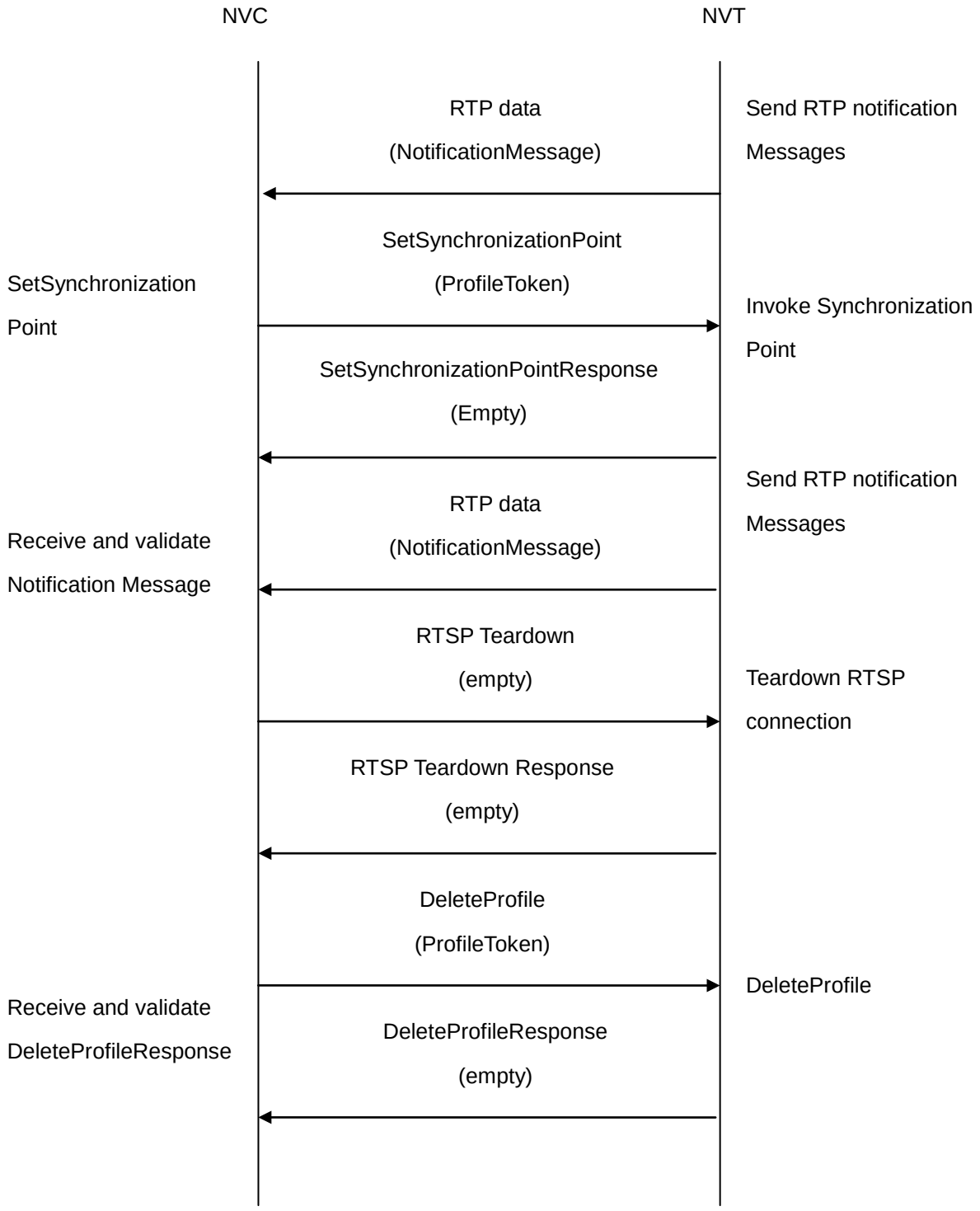
Pre-Requisite: The device needs to provide at least one topic representing a certain property. If the device does not support a property event the vendor MUST make sure that another event is sent during testing.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
CreateProfile
(name="Test" token="Test")
CreatesProfile
Receive and validate CreateProfile Response
CreateProfile
(ProfileToken="Test")
GetMetadataConfigurations
(empty)
Receive and validate GetMetadata ConfigurationsResponse
GetMetadataConfigurationsResponse
(MetadataConfigurations)
GetVideoSourceConfigurations
(empty)
Receive and validate VideoSource ConfigurationsResponse
GetVideoSourceConfigurationsResponse
(VideoSourceConfigurations)
AddVideoSourceConfiguration
(ProfileToken="Test",VideoSourceToken)
Add VideoSource Configuration to created profile
Receive and validate AddVideoSource ConfigurationResponse
AddVideoSourceConfigurationResponse
(empty)
AddMetadataConfiguration
(ProfileToken="Test",MetadataToken)
Add Metadata Configuration to created profile
Receive and validate AddMetadata ConfigurationResponse
AddMetadataConfigurationResponse
(empty)


NVC
NVT
SetMetadataConfiguration
(MetdataConfiguration)
Configure Metadata Configuration
SetMetadataConfigurationresponse
(empty)
GetStreamUri
(ProfileToken, StreamSetup)
Receive and validate GetStreamUriResponse
GetStreamUriResponse
(StreamUri)
RTSP Describe
(empty)
Invoke RTSP Describe, parse sdp file
RTSP Describe Response
(sdp file)
RTSP Setup
(empty)
RTSP Setup for Metadatastream
RTSP Setup Response
(empty)
RTSP Play
(empty)
RTSP Play for Metadatastream
RTSP Play Response
(empty)



Test Procedure:

1. Start an NVC.
2. Start an NVT.

3. NVC invokes CreateProfile (name="Test", ProfileToken="Test") to create a new empty profile that is used for this test scenario.
4. Verify that the DUT sends a valid CreateProfileResponse
5. NVC will invoke GetMetadataConfigurations to retrieve all existing MetadataConfigurations of the device
6. Verify that the DUT sends a valid GetMetadataConfigurationResponse (that contains at least one MetadataConfiguration).
7. NVC will invoke GetVideoSourceConfigurations
8. Verify that the DUT sends a valid GetVideoSourceConfigurationResponse
9. NVC will select a VideoSourceConfiguration and add this configuration to the created profile
10. Verify that the DUT sends a valid AddVideoSourceConfigurationResponse
11. NVC will select a MetadataConfiguration and add this configuration to the created profile
12. Verify that the DUT sends a valid AddMetadataConfigurationResponse
13. NVC will invoke SetMetadataConfiguration(<Analytics>false</Analytics>,<Events/>) to configure the Metadatastream; The NVC is interested in receiving all events, therefore no Filter is applied. For details on the usage of the MetadataConfiguration elements, see Appendix A.16.
14. Verify that the DUT sends a valid SetMetadataConfigurationResponse
15. NVC will invoke GetStreamUri (ProfileToken, RTP-Unicast)
16. Verify that the DUT sends a GetStreamUriResponse including a valid StreamUri
17. NVC will invoke RTSP Describe to retrieve the sdp file
18. Verify that DUT sends a 200 OK Response
19. Validate sdp file (sdp file contains only one media section; rtpmap=vnd.onvif.metadata)
20. NVC will invoke RTSP Setup for the Metadatastream
21. Verify that the DUT send a 200 OK Response
22. NVC will invoke RTSP Play
23. Verify that the DUT sends a 200 OK Response
24. Receive and validate RTP Notification messages
25. NVC will invoke the SetSynchronizationPoint command to trigger events; if the device does not support property events the vendor MUST trigger the events manually.
26. Validate that DUT sends a valid SetSynchronizationPointResponse
27. Verify that at least one RTP Notification is sent.
28. Receive and validate RTP Notification messages and check that the PropertyOperation is "Initialized" or "Changed" if it is an Property event
29. NVC will invoke RTSP Teardown to terminate the RTSP session

30. Verify that DUT sends a 200 OK Response
31. NVC will invoke DeleteProfile
32. Verify that DUT sends a DeleteProfileResponse

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send a CreateProfileResponse

The DUT did not send a valid CreateProfileResponse

The DUT did not send valid GetMetadataConfigurationsResponse; At least one MetadataConfiguaration MUST be present

The DUT did not send a valid GetVideoSourceConfigurationsResponse

The DUT did not send a valid AddVideoSourceConfigurationResponse

The DUT did not send a valid AddMetadataConfigurationResponse

The DUT did not send a valid SetMetadataConfigurationResponse

The DUT did not send a GetStreamUriResponse including a valid StreamUri

The DUT did not send a 200 OK RTSP DESCRIBE Response

The DUT did not send a valid sdp file

The DUT did not send a 200 OK RTSP SETUP Response

The DUT did not send a 200 OK RTSP PLAY Response

The DUT did not send RTP data

The DUT did not send a SetSynchronizationPointResponse

The DUT did not send at least one event

The DUT did not send RTP data with PropertyOperation=”Initialized” or “Changed” if it is an property event

The DUT did not send a 200 OK RTSP TEARDOWN Response

The DUT did not send a DeleteProfileResponse

10 PTZ Control Test Cases

10.1 PTZ Node

10.1.1 NVT PTZ NODES

Test Label: NVT PTZ Nodes Validation

ONVIF Core Specification Coverage: 13.2.1 GetNodes

Device Type: NVT

Command Under Test: GetNodes

WSDL Reference: ptz.wsdl

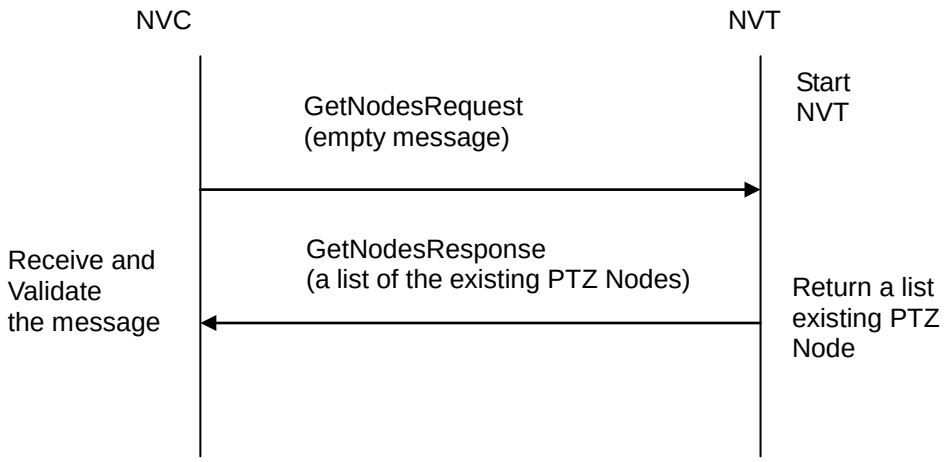
Requirement Level: MUST IF SUPPORTED (PTZ)

Test Purpose: To verify GetNodes command.

Pre-Requisite: PTZ is supported by NVT, and NVC gets the ptz service entry point by GetCapabilities command.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNodesRequest message to retrieve the list of PTZ nodes supported by NVT.
4. Verify that the NVT returns at-least one PTZNode in the GetNodesResponse message.

5. Validate PTZNodes of GetNodesResponse message (check mandatory element of SupportedPTZSpaces, MaximumNumberOfPresets, and HomeSupported.)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetNodesResponse message.

The DUT did not send valid GetNodesResponse message.

The DUT did not send GetNodesResponse message with at-least one PTZNode.

10.1.2 NVT PTZ NODE

Test Label: NVT PTZ Node Validation

ONVIF Core Specification Coverage: 13.2.2 GetNode

Device Type: NVT

Command Under Test: GetNode

WSDL Reference: ptz.wsdl

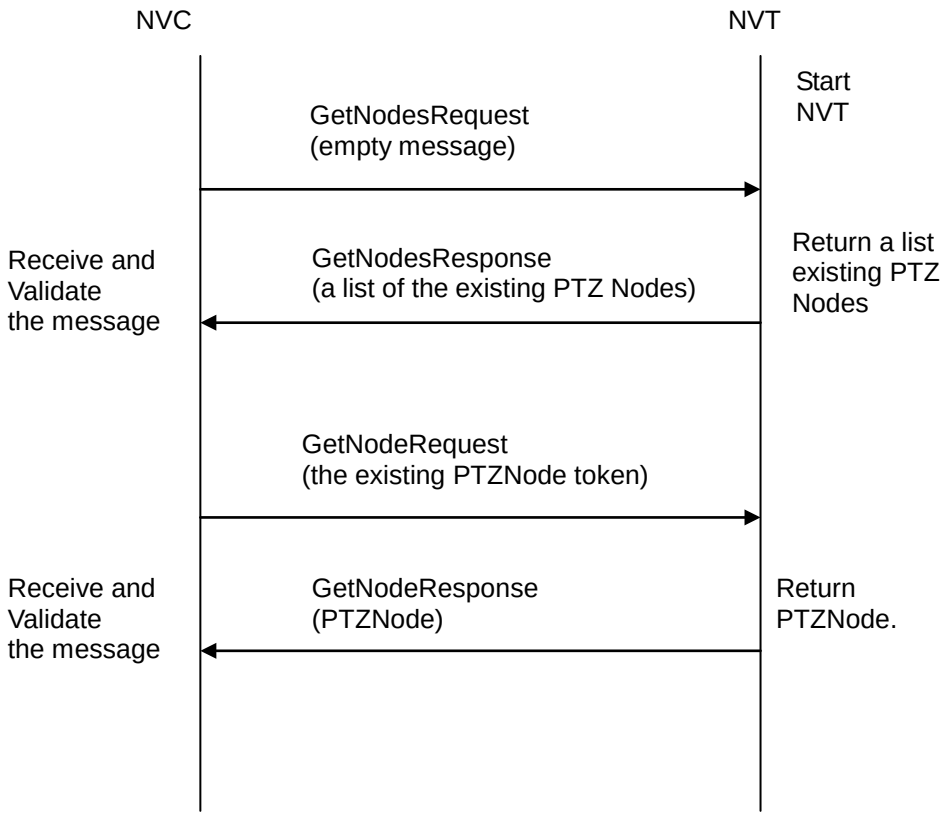
Requirement Level: MUST IF SUPPORTED (PTZ)

Test Purpose: To verify GetNode command.

Pre-Requisite: PTZ is supported by NVT, and NVC gets the ptz service entry point by GetCapabilities command.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNodesRequest message to retrieve a list of the existing PTZNodes.
4. Verify that NVT returns at-least one PTZNode in the GetNodesResponse message.
5. NVC will invoke GetNodeRequest message (NodeToken of existing PTZNode) to retrieve the specific PTZNode
6. Verify that NVT returns a PTZNode in GetNodeResponse message.
7. Validate PTZNode of GetNodeResponse message (check mandatory element of SupportedPTZSpaces, MaximumNumberOfPresets, and HomeSupported.)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetNodesResponse message.

The DUT did not send valid GetNodesResponse message.

The DUT did not send GetNodesResponse message with at-least one PTZNode.

The DUT did not send GetNodeResponse message.

The DUT did not send valid GetNodeResponse message.

10.1.3 NVT SOAP FAULT MESSAGE

Test Label: NVT PTZ Soap Fault Message for Invalid GetNode Request Message

ONVIF Core Specification Coverage: 13.2.2 GetNode

Device Type: NVT

Command Under Test: GetNode

WSDL Reference: ptz.wsdl

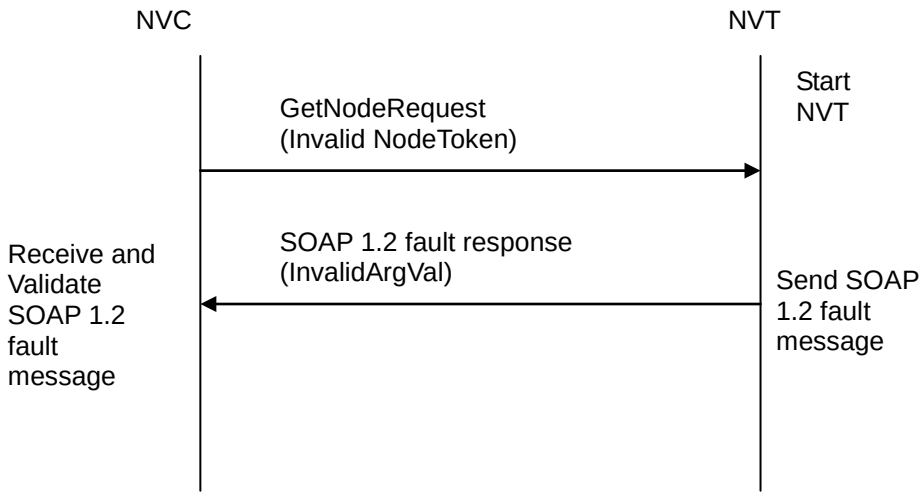
Requirement Level: SHOULD IF SUPPORTED (PTZ)

Test Purpose: To verify that NVT generates a SOAP fault message to invalid GetNode message.

Pre-Requisite: PTZ is supported by NVT, and NVC gets the ptz service entry point by GetCapabilities command.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetNodeRequest message with invalid NodeToken (not NodeToken of existing PTZNode. example **NodeToken** ReferenceToken = **"xyz"**).
4. Verify the NVT generates a SOAP 1.2 fault message (**InvalidArgVal/NoEntity**).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct SOAP 1.2 fault message (fault code, namespace etc).

10.2 PTZ Configuration

10.2.1 NVT PTZ CONFIGURATIONS

Test Label: NVT PTZ Configurations Validation

ONVIF Core Specification Coverage: 13.3.1 GetConfigurations

Device Type: NVT

Command Under Test: GetConfigurations

WSDL Reference: ptz.wsdl

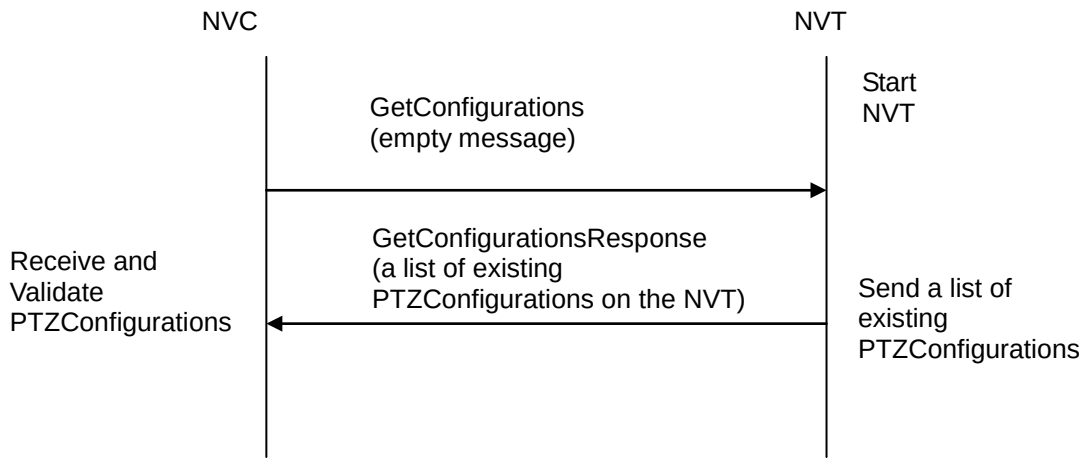
Requirement Level: MUST IF SUPPORTED (PTZ)

Test Purpose: To retrieve NVT PTZ Configurations setting.

Pre-Requisite: PTZ is supported by NVT, and NVC gets the ptz service entry point by GetCapabilities command.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetConfigurationsRequest message to retrieve a list of existing PTZConfigurations on the NVT.
4. Verify that the NVT returns at-least one PTZConfiguration in the GetConfigurationsResponse message.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetConfigurationsResponse message.

The DUT did not send valid GetConfigurationsResponse message.

The DUT did not send GetConfigurationsResponse message with at least one PTZConfiguration.

10.2.2 NVT PTZ CONFIGURATION

Test Label: NVT PTZ Configuration Validation

ONVIF Core Specification Coverage: 13.3.2 GetConfiguration

Device Type: NVT

Command Under Test: GetConfiguration

WSDL Reference: ptz.wsdl

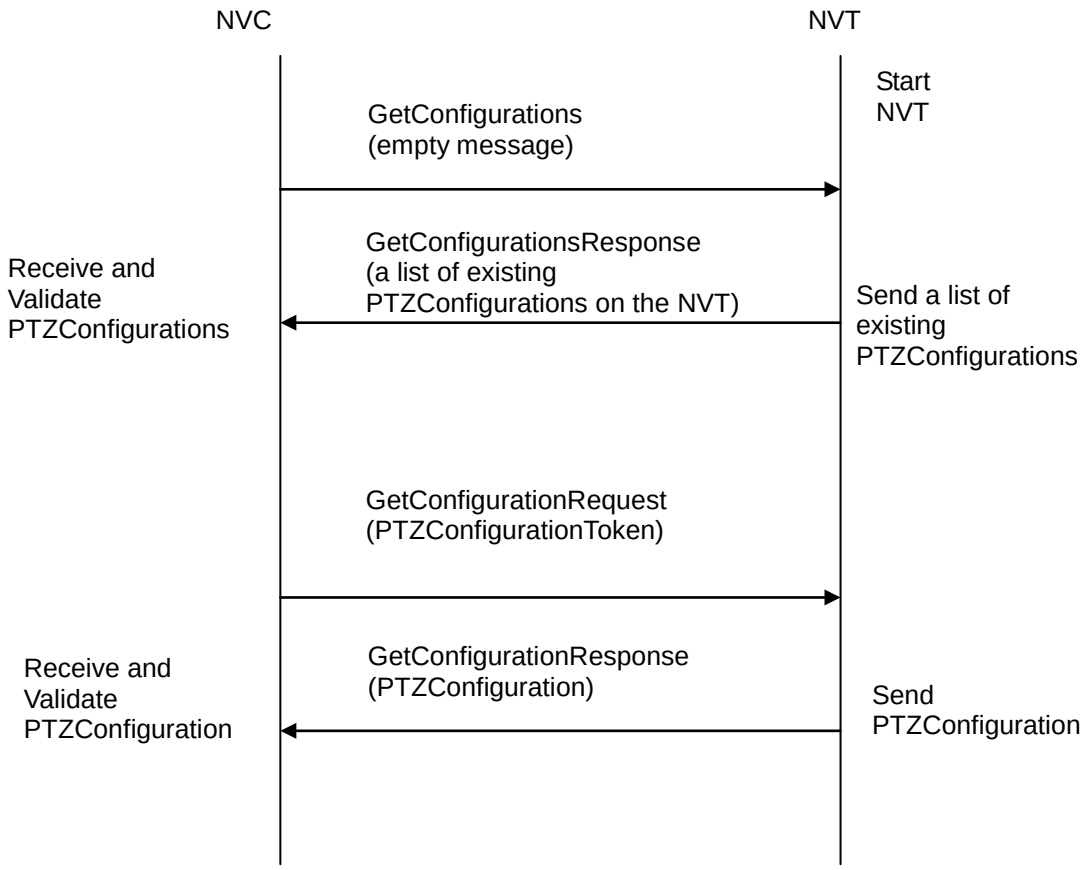
Requirement Level: MUST IF SUPPORTED (PTZ)

Test Purpose: To retrieve NVT PTZ Configuration setting.

Pre-Requisite: PTZ is supported by NVT, and NVC gets the ptz service entry point by GetCapabilities command.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetConfigurationsRequest message to retrieve a list of existing PTZConfigurations.
4. Verify the GetConfigurationsResponse from NVT (a list of existing PTZConfigurations).
5. NVC will invoke GetConfigurationRequest message (PTZConfigurationToken of existing PTZConfiguration) to retrieve requested PTZConfiguration.
6. Verify the GetConfigurationResponse from NVT (PTZConfiguration includes a NodeToken, and at least one parameter (DefaultAbsolutePantTiltPositionSpace, DefaultAbsoluteZoomPositionSpace, DefaultRelativePanTiltTranslationSpace, DefaultRelativeZoomTranslationSpace, DefaultContinuousPanTiltVelocitySpace, DefaultContinuousZoomVelocitySpace, DefaultPTZSpeed, DefaultPTZTimeout, PanTiltLimits, and ZoomLimits)).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetConfigurationsResponse message.

The DUT did not send valid GetConfigurationsResponse message.

The DUT did not send GetConfigurationsResponse message with at least one PTZConfiguration.

The DUT did not send GetConfigurationResponse message.

The DUT did not send valid GetConfigurationResponse message.

The DUT did not send GetConfigurationResponse message with NodeToken.

The DUT did not send GetConfigurationResponse message with at least one parameter (excluding NodeToken).

10.2.3 NVT PTZ CONFIGURATION OPTIONS

Test Label: NVT PTZ Configuration Options Validation

ONVIF Core Specification Coverage: 13.3.3 GetConfigurationOptions

Device Type: NVT

Command Under Test: GetConfigurationOptions

WSDL Reference: ptz.wsdl

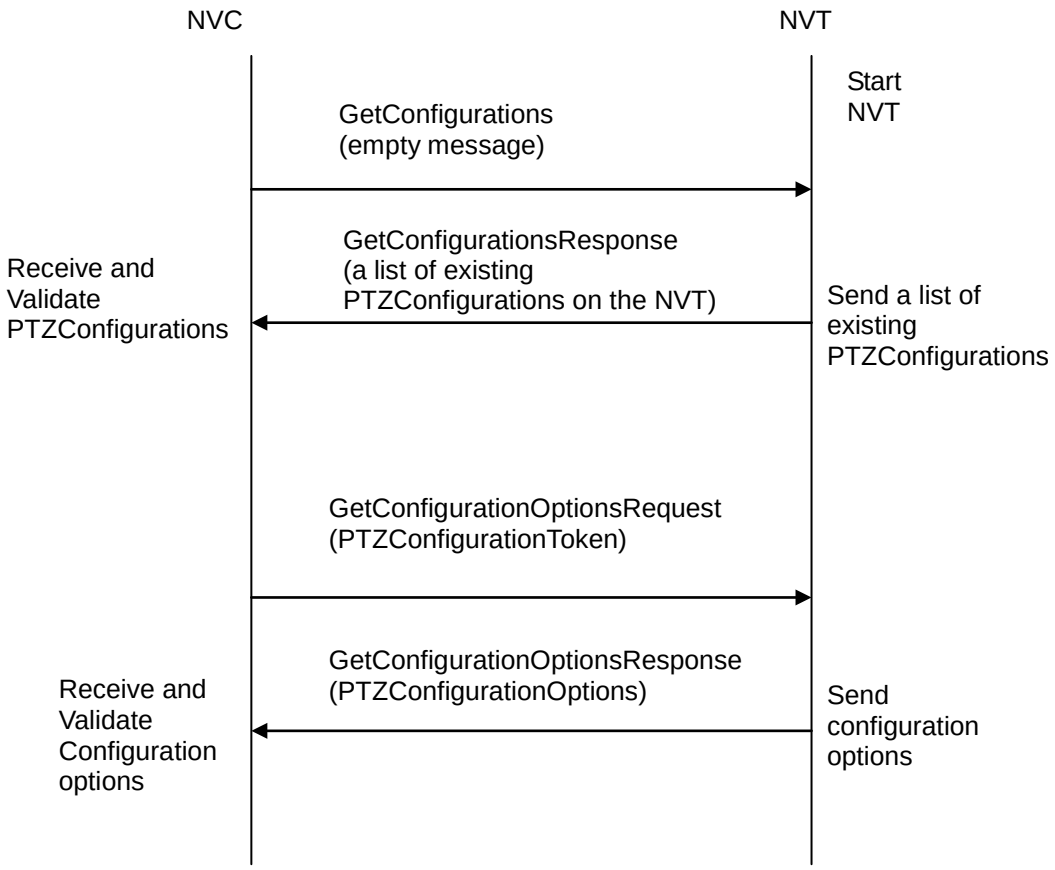
Requirement Level: MUST IF SUPPORTED (PTZ)

Test Purpose: To retrieve NVT PTZ Configuration Options setting.

Pre-Requisite: PTZ is supported by NVT, and NVC gets the ptz service entry point by GetCapabilities command.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetConfigurationsRequest message to retrieve a list of existing PTZConfigurations on the NVT.
4. Verify the GetConfigurationsResponse from NVT (a list of existing PTZConfiguration).
5. NVC will invoke GetConfigurationOptionsRequest message to retrieve PTZConfigurationOptions.
6. Verify the GetConfigurationOptionsResponse from NVT (valid Spaces and PTZTimeout).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetConfigurationsResponse message.

The DUT did not send valid GetConfigurationsResponse message.

The DUT did not send GetConfigurationsResponse message with at least one PTZConfiguration.

The DUT did not send GetConfigurationOptionsResponse message.

The DUT did not send valid GetConfigurationOptionsResponse message.

The DUT did not send GetConfigurationOptionsResponse message with valid Spaces and PTZTimeout.

10.2.4 NVT PTZ SET CONFIGURATION

Test Label: NVT PTZ Configurations

ONVIF Core Specification Coverage: 13.3.4 SetConfiguration

Device Type: NVT

Command Under Test: SetConfiguration

WSDL Reference: ptz.wsdl

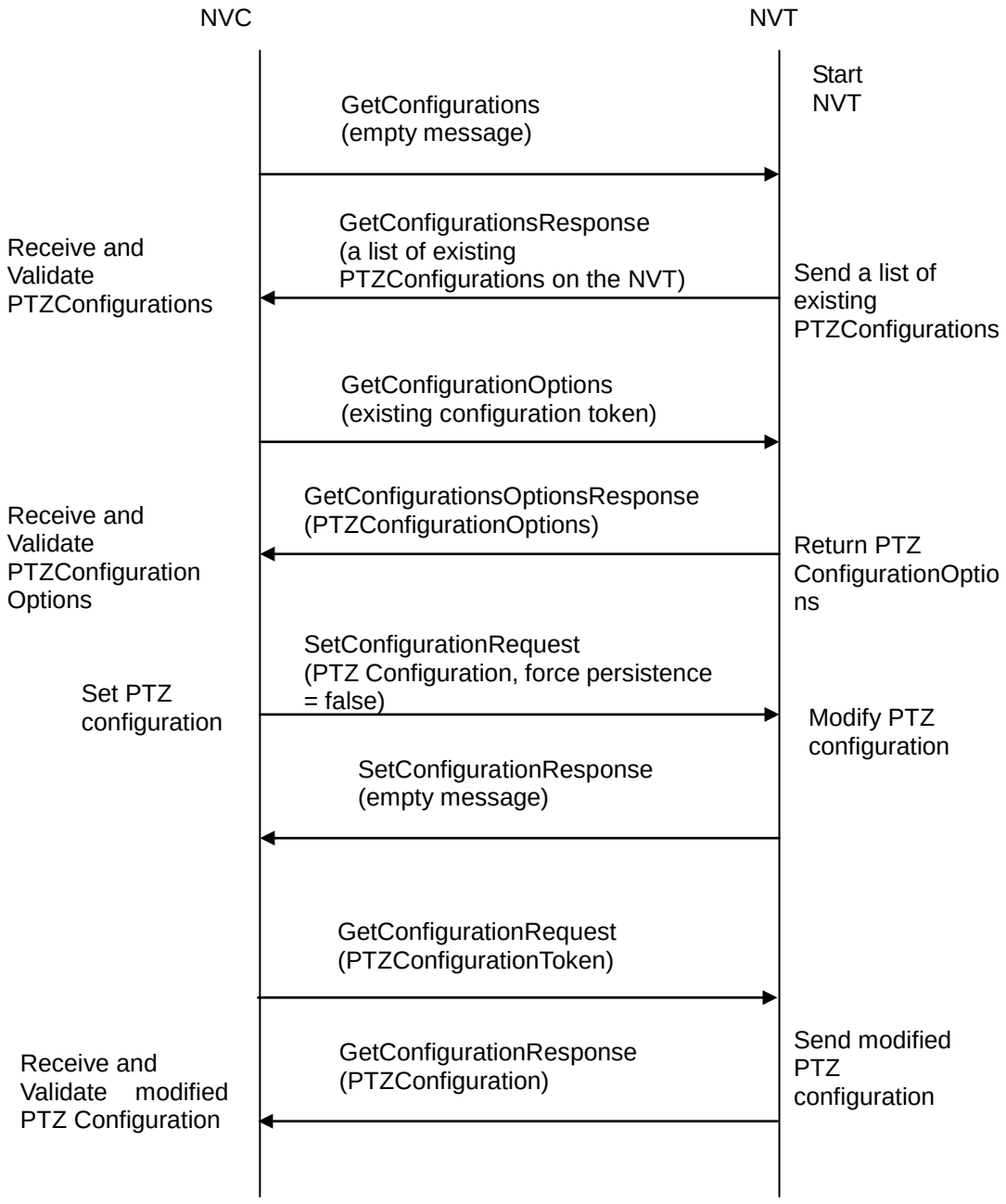
Requirement Level: MUST IF SUPPORTED (PTZ)

Test Purpose: To verify NVT PTZ Configuration Setting

Pre-Requisite: PTZ is supported by NVT, and NVC gets the ptz service entry point by GetCapabilities command.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke GetConfigurationsRequest message to retrieve a list of existing PTZConfigurations.
4. Verify the GetConfigurationsResponse from NVT (a list of existing PTZConfigurations).

5. NVC will invoke **GetConfigurationOptions** message (**ConfigurationToken of existing PTZConfiguration**) to retrieve the range of **PTZTimeout** that can be changed.
6. Verify that NVT returns **PTZConfigurationOptions** in **GetConfigurationOptionsResponse** message.
7. NVC will invoke SetConfigurationRequest message (**DefaultPTZTimeout = [Max or Min of duration value]**, and force persistence = false). DefaultPTZTimeout will be set to Max of the duration value. If DefaultPTZTimeout of DUT is same value with Max of duration value, this value will be set to Min of the duration value.
8. NVT modifies PTZ Configuration and return with SetConfigurationResponse message indicating success.
9. NVC will verify the modified PTZ configuration by invoking GetConfigurationRequest message
10. Verify that NVT returns the modified PTZ Configuration in the GetConfigurationResponse message (**DefaultPTZTimeout = [Max or Min of the duration value]**).

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetConfigurationsResponse message.

The DUT did not send valid GetConfigurationsResponse message.

The DUT did not send GetConfigurationsResponse message with at least one PTZConfiguration.

The DUT did not send GetConfigurationOptionsResponse message.

The DUT did not send valid GetConfigurationOptionsResponse message.

The DUT did not send SetConfigurationResponse message.

The DUT did not send GetConfigurationResponse message.

The DUT did not send valid GetConfigurationResponse message.

The DUT did not modify PTZConfiguration by requested SetConfigurationRequest message.

10.2.5 NVT SOAP FAULT MESSAGE

Test Label: NVT PTZ Soap Fault Message for invalid SetConfiguration Request message

ONVIF Core Specification Coverage: 13.3.4 SetConfiguration

Device Type: NVT

Command Under Test: SetConfiguration

WSDL Reference: ptz.wsdl

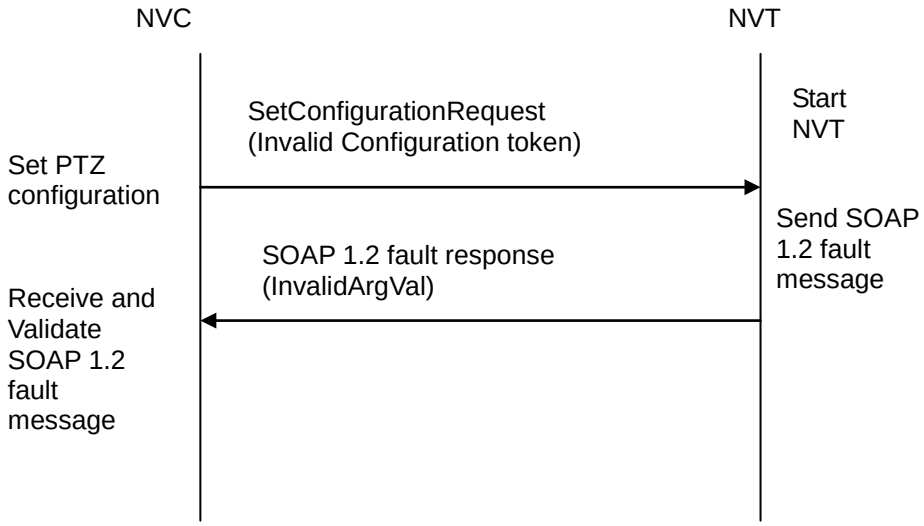
Requirement Level: SHOULD IF SUPPORTED (PTZ)

Test Purpose: To verify that NVT generates a SOAP fault message if an invalid PTZ Configuration message is sent.

Pre-Requisite: PTZ is supported by NVT, and NVC gets the ptz service entry point by GetCapabilities command.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC will invoke SetConfigurationRequest message with an invalid Configuration token.
4. NVT will generate a SOAP 1.2 fault message (**InvalidArgVal/NoConfig**)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send SOAP 1.2 fault message.

The DUT did not send correct SOAP 1.2 fault message (fault code, namespace etc).

10.3 Move Operation

10.3.1 NVT PTZ ABSOLUTE MOVE

Test Label: NVT PTZ Absolute Move Operation

ONVIF Core Specification Coverage: 13.4.1 AbsoluteMove

Device Type: NVT

Command Under Test: AbsoluteMove

WSDL Reference: ptz.wsdl

Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Absolute Move)

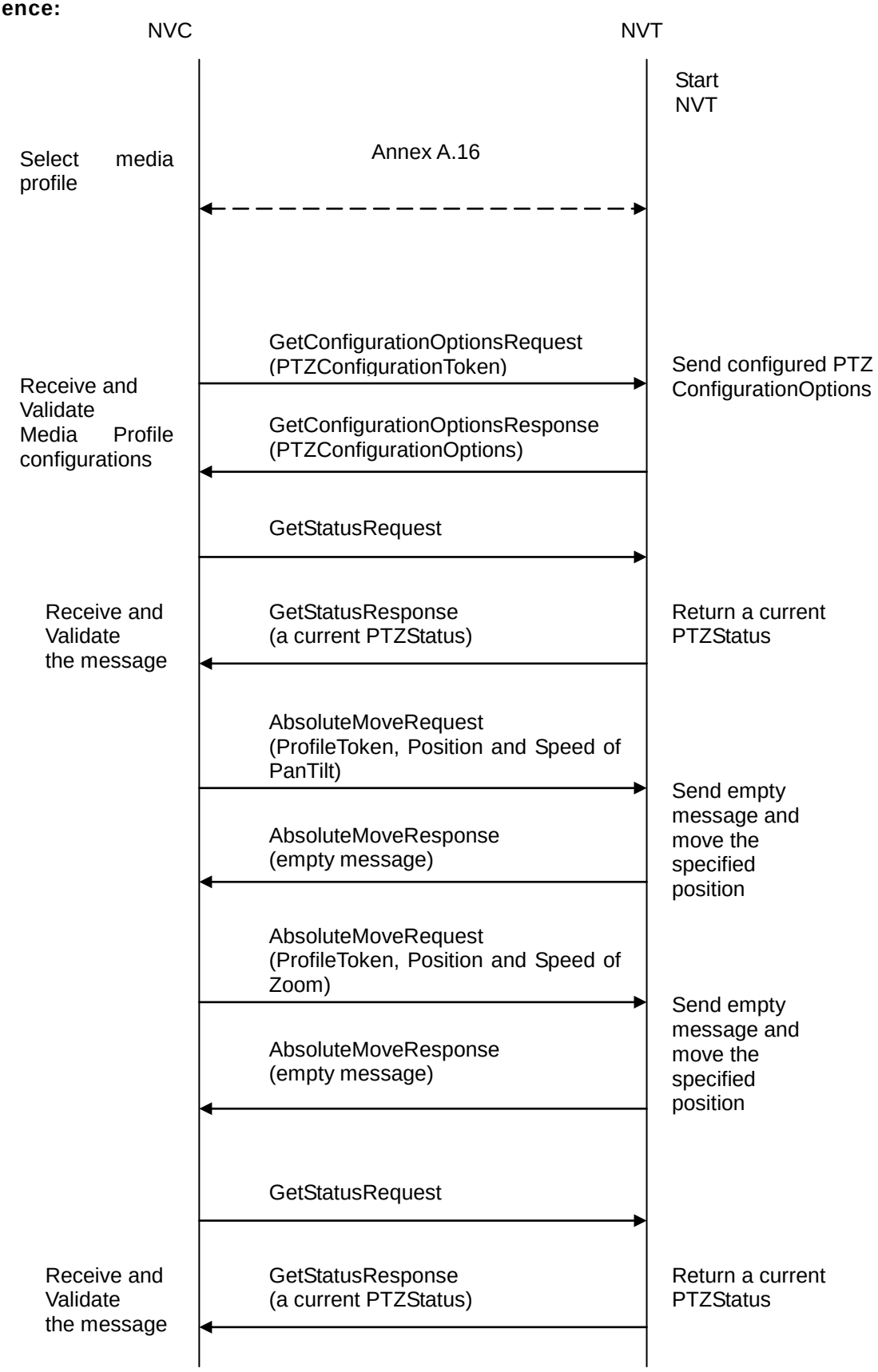
Test Purpose: To verify NVT PTZ AbsoluteMove operation

Pre-Requisite: PTZ is supported by NVT, and a function of Absolute movements is implemented. In addition, NVC gets the ptz service entry point by GetCapabilities command.

A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC configures and selects a media profile as described in Annex A.16.
4. NVC will invoke GetConfigurationOptionsRequest message to retrieve PTZConfigurationOptions.
5. Verify that NVT returns GetConfigurationOptionsResponse with valid Spaces and PTZTimeout, and has the function of Absolute movement.
6. NVC will invoke GetStatusRequest message to get a current PTZStatus.
7. NVT returns a current PTZStatus in the GetStatusResponse.
8. If Absolute move is supported for 'Pan Tilt',, NVC will invoke AbsoluteMoveRequest message (**ProfileToken, Position:PanTilt = ["x", "y"], Speed:PanTilt=[“x”, “y”]**). The Speed:PanTilt parameter is added if supported Speed:PanTilt.
9. If NVC invoked AbsoluteMoveRequest message for PanTilt, verify that NVT returns AbsoluteMoveResponse message indicating success.
10. If Absolute move is supported for 'Zoom', NVC will invoke AbsoluteMoveRequest message (**ProfileToken, Position:Zoom = ["x"], Speed:Zoom = ["x"]**). The Speed:Zoom parameter is added if supported Speed:Zoom.
11. If NVC invoked AbsoluteMoveRequest message for Zoom, verify that NVT returns AbsoluteMoveResponse message indicating success.
12. NVC will invoke GetStatusRequest message to get a current PTZStatus.
13. Verify that NVT moves to the specified position by GetStatusResponse message.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetConfigurationOptionsResponse message.

The DUT did not send valid GetConfigurationOptionsResponse message.

The DUT did not send GetConfigurationOptionsResponse message with valid Spaces and PTZTimeout.

The DUT did not send GetStatusResponse message.

The DUT did not send valid GetStatusResponse message.

The DUT did not send AbsoluteMoveResponse message.

The DUT did not send GetStatusResponse message with the specified position after moved by NVC.

Note: If NVT does not return a current position by GetStatusResponse, the specified position after moved by NVC isn't checked by NVC.

PTZ accuracy is out of scope for this Test Specification. Therefore the position reported by the NVT in the GetStatusResponse does not have to be exactly the same as the position requested by the NVC in the AbsoluteMoveRequest.

10.3.2 NVT SOAP FAULT MESSAGE

Test Label: PTZ NVT PTZ Soap Fault Message for Invalid AbsoluteMove Request Message

ONVIF Core Specification Coverage: 13.4.1 AbsoluteMove

Device Type: NVT

Command Under Test: AbsoluteMove

WSDL Reference: ptz.wsdl

Requirement Level: SHOULD IF SUPPORTED (PTZ) & IMPLEMENTED (Absolute Movements)

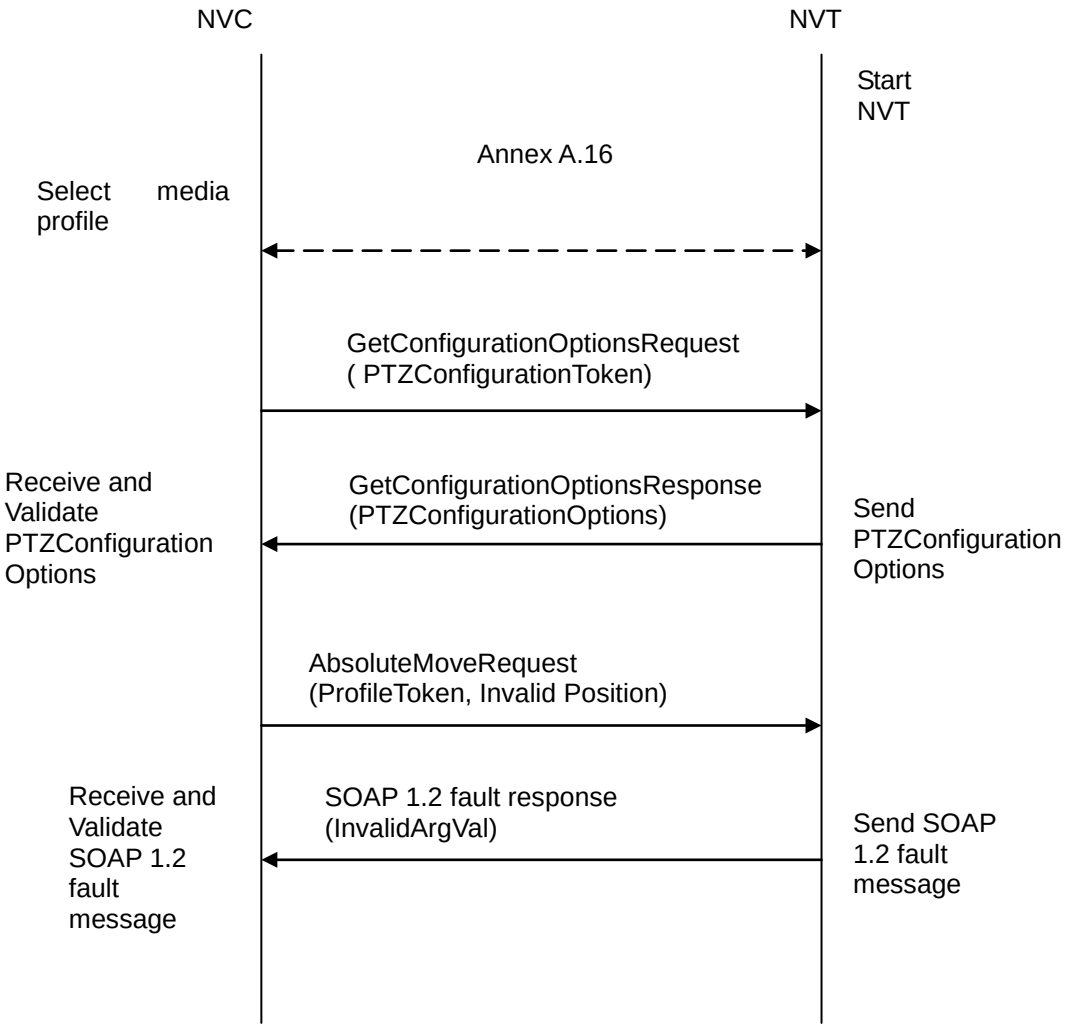
Test Purpose: To verify that NVT generates a SOAP fault message to AbsoluteMove operation with out of bounds values.

Pre-Requisite: PTZ is supported by NVT, and a function of Absolute movements is implemented. In addition, NVC gets the ptz service entry point by GetCapabilities command.

A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC configures and selects a media profile as described in Annex A.16.
4. NVC will invoke GetConfigurationOptionsRequest message (PTZConfigurationToken,).
5. NVT returns existing PTZConfiguration in the GetConfigurationOptionsResponse message.
6. NVC will invoke AbsoluteMoveRequest message (ProfileToken, PanTilt = ["x(Out of range)", "y(Out of range)"]).
7. Verify the NVT generates a SOAP 1.2 fault message (**InvalidArgVal/InvalidPosition**)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetConfigurationOptionsResponse message.

The DUT did not send valid GetConfigurationOptionsResponse message.

The DUT did not send SOAP 1.2 fault message against AbsoluteMoveRequest message.

The DUT did not send correct SOAP 1.2 fault message (fault code, namespace etc) against AbsoluteMoveRequest message.

10.3.3 NVT PTZ RELATIVE MOVE

Test Label: NVT PTZ Relative Move Operation

ONVIF Core Specification Coverage: 13.4.2 RelativeMove

Device Type: NVT

Command Under Test: RelativeMove

WSDL Reference: ptz.wsdl

Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Relative Move)

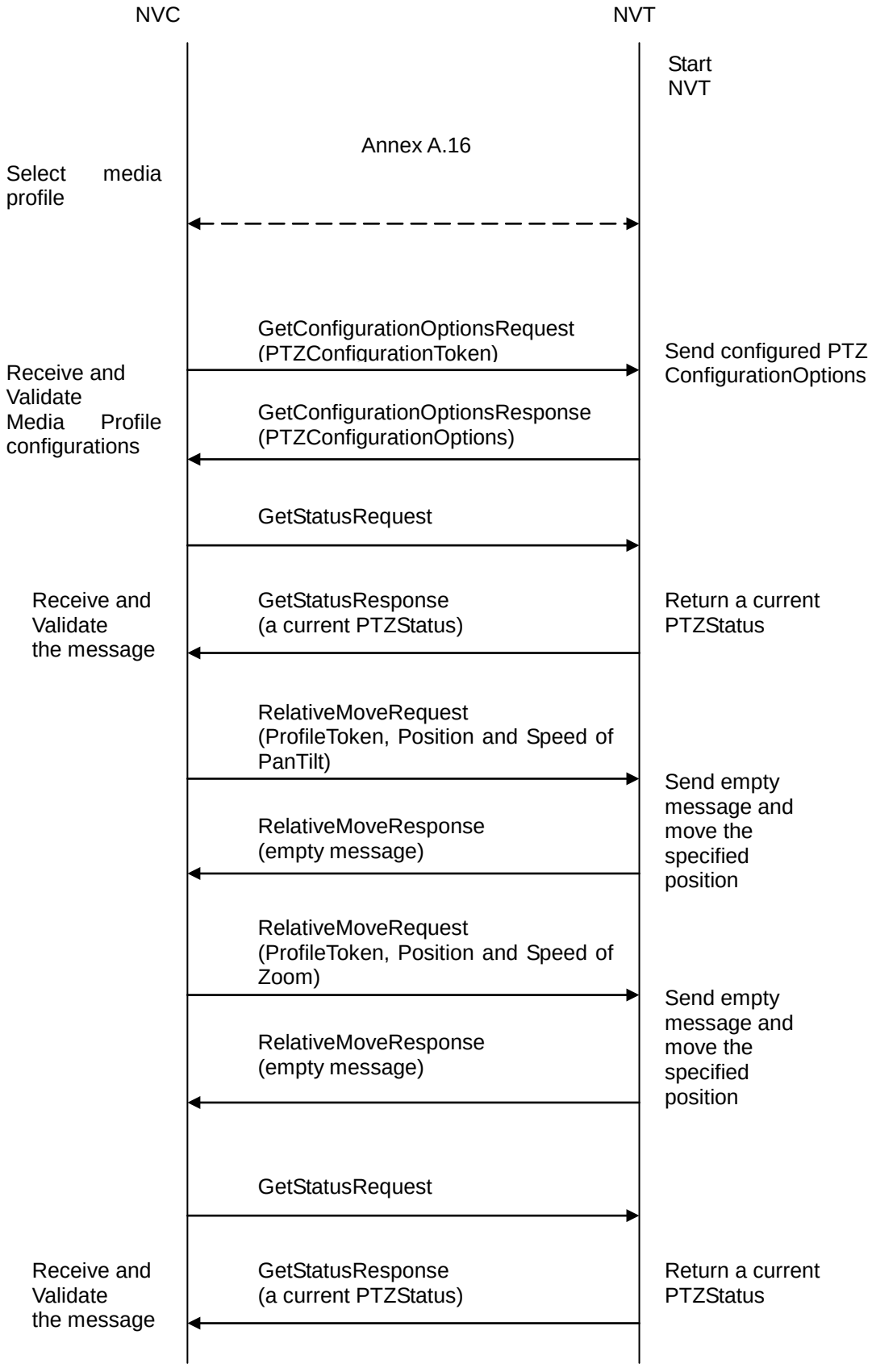
Test Purpose: To verify NVT PTZ RelativeMove operation

Pre-Requisite: PTZ is supported by NVT, and a function of Relative movements is implemented. In addition, NVC gets the ptz service entry point by GetCapabilities command.

A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC configures and selects a media profile as described in Annex A.16.
4. NVC will invoke GetConfigurationOptionsRequest message to retrieve PTZConfigurationOptions.
5. Verify that NVT returns GetConfigurationOptionsResponse with valid Spaces and PTZTimeout, and has the function of Relative movement.
6. NVC will invoke GetStatusRequest message to get a current PTZStatus.
7. NVT returns a current PTZStatus in the GetStatusResponse.
8. If PanTilt of Relative movement is supported (there is a parameter of RelativePanTiltTranslationSpace in PTZConfigurationOptions), NVC will invoke RelativeMoveRequest message (**ProfileToken, Position:PanTilt = ["x", "y"], Speed:PanTilt=[“x”, “y”]**). The Speed:PanTilt parameter is added if supported Speed:PanTilt.
9. If NVC invoked RelativeMoveRequest message for PanTilt, verify that NVT returns RelativeMoveResponse message indicating success.
10. If Zoom Relative movement is supported (there is a parameter of RelativeZoomTranslationSpace in PTZConfigurationOptions), NVC will invoke RelativeMoveRequest message (**ProfileToken, Position:Zoom = ["x"], Speed:Zoom = ["x"]**). The Speed:Zoom parameter is added if supported Speed:Zoom.
11. If NVC invoked RelativeMoveRequest message for Zoom, verify that NVT returns RelativeMoveResponse message indicating success.
12. NVC will invoke GetStatusRequest message to get a current PTZStatus.
13. Verify that NVT moves to the specified position by GetStatusResponse message.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetConfigurationOptionsResponse message.

The DUT did not send valid GetConfigurationOptionsResponse message.

The DUT did not send GetConfigurationOptionsResponse message with valid Spaces and PTZTimeout.

The DUT did not send GetStatusResponse message.

The DUT did not send valid GetStatusResponse message.

The DUT did not send RelativeMoveResponse message.

The DUT did not send GetStatusResponse message with the specified position after moved by NVC.

Note: If NVT does not return a current position by GetStatusResponse, the specified position after moved by NVC isn't checked by NVC.

PTZ accuracy is out of scope for this Test Specification. Therefore the position reported by the NVT in the GetStatusResponse does not have to be exactly the same as the position requested by the NVC in the RelativeMoveRequest.

10.3.4 NVT PTZ CONTINUOUS MOVE

Test Label: NVT PTZ Continuous Move Operation

ONVIF Core Specification Coverage: 13.4.3 ContinuousMove

Device Type: NVT

Command Under Test: ContinuousMove

WSDL Reference: ptz.wsdl

Requirement Level: MUST IF SUPPORTED (PTZ)

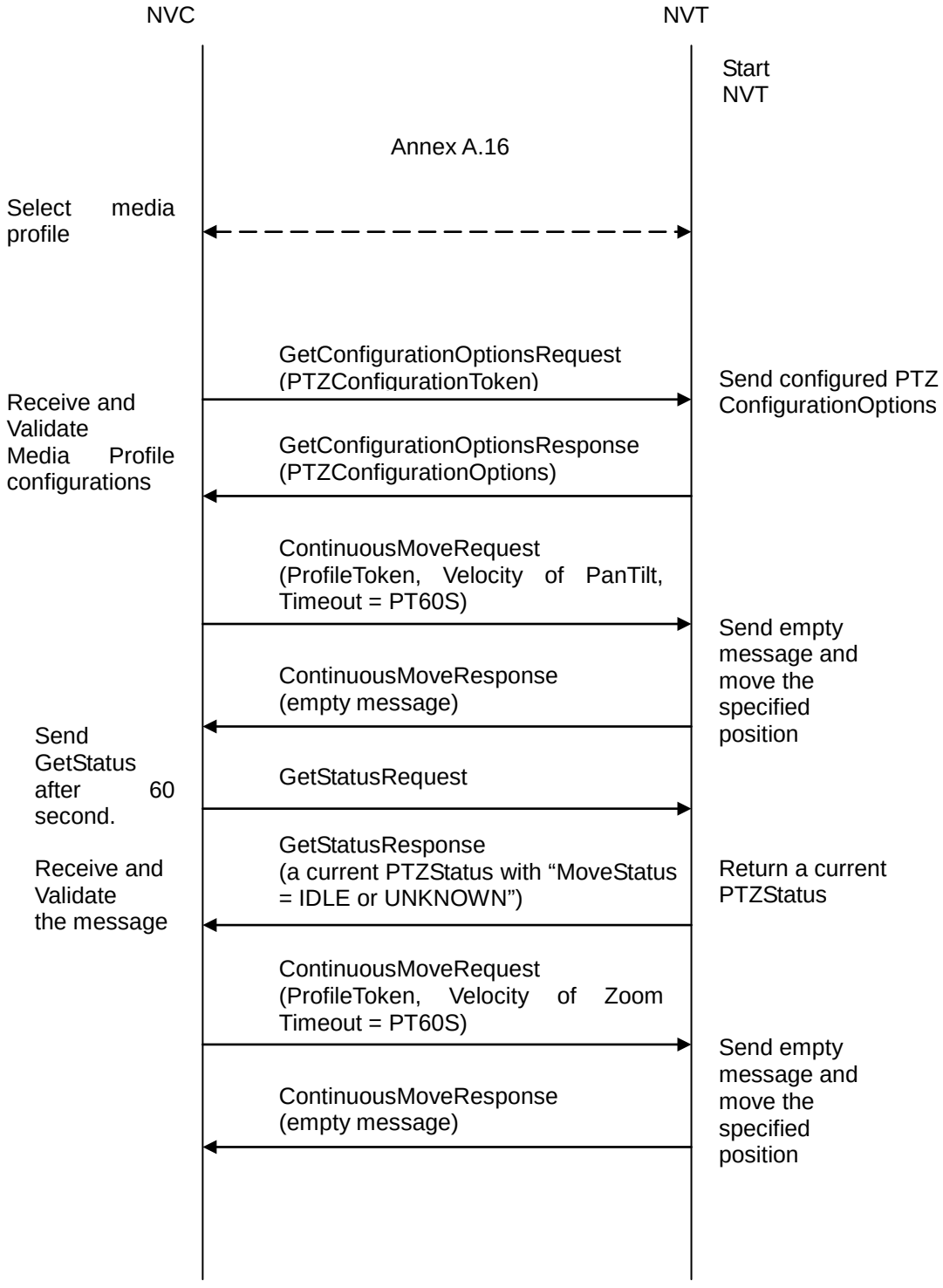
Test Purpose: To verify NVT PTZ ContinuousMove operation with timeout parameter

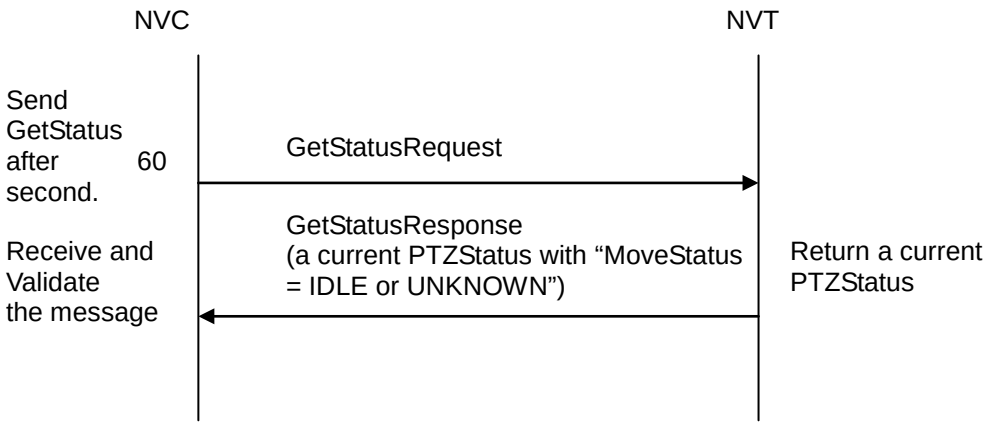
Pre-Requisite: PTZ is supported by NVT, and a function of Continuous movements is implemented. In addition, NVC gets the ptz service entry point by GetCapabilities command.

A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:





Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC configures and selects a media profile as described in Annex A.16.
4. NVC will invoke GetConfigurationOptionsRequest message to retrieve PTZConfigurationOptions.
5. Verify that NVT returns GetConfigurationOptionsResponse with valid Spaces and PTZTimeout, and has the function of Relative movement.
6. If PanTilt of Continuous movement is supported (there is a parameter of ContinuousPanTiltVelocitySpace in PTZConfigurationOptions), NVC will invoke ContinuousMoveRequest message (ProfileToken, Velocity:PanTilt = ["x", "y"], Timeout = PT60S).
7. If NVC invoked ContinuousMoveRequest message for PanTilt, verify that NVT returns ContinuousMoveResponse message indicating success.
8. If NVC invoked ContinuousMoveRequest message for PanTilt, NVC will invoke GetStatusRequest message to get a current PTZStatus after 60 second.
9. If NVC invoked ContinuousMoveRequest message for PanTilt, verify that the NVT returns GetStatusResponse with “MoveStatus = IDLE or UNKNOWN”.
10. If Zoom of Continuous movement is supported (there is a parameter of ContinuousZoomVelocitySpace in PTZConfigurationOptions), NVC will invoke ContinuousMoveRequest message (ProfileToken, Velocity:Zoom = ["x"], Timeout = PT60S).
11. If NVC invoked ContinuousMoveRequest message for Zoom, verify that NVT returns ContinuousMoveResponse message indicating success.
12. If NVC invoked ContinuousMoveRequest message for Zoom, NVC will invoke GetStatusRequest message to get a current PTZStatus after 60 second.
13. If NVC invoked ContinuousMoveRequest message for PanTilt, verify that the NVT returns GetStatusResponse with “MoveStatus = IDLE or UNKNOWN”.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetConfigurationOptionsResponse message.

The DUT did not send valid GetConfigurationOptionsResponse message.

The DUT did not send GetConfigurationOptionsResponse message with valid Spaces and PTZTimeout.

The DUT did not send GetStatusResponse message.

The DUT did not send valid GetStatusResponse message.

The DUT did not send ContinuousMoveResponse message.

The DUT did not send GetStatusResponse message with “MoveStatus = MOVING or UNKNOWN” after executing Test Procedure 9 and 15.

The DUT did not send GetStatusResponse message with “MoveStatus = IDLE or UNKNOWN” after executing Test Procedure 11 and 17.

Note: If NVT does not return a current MoveStatus by GetStatusResponse, the MoveStatus isn’t checked by NVC.

10.3.5 NVT PTZ CONTINUOUS MOVE & STOP

Test Label: NVT PTZ Continuous Move and Stop Operation

ONVIF Core Specification Coverage: 13.4.3 ContinuousMove, 13.4.4 Stop

Device Type: NVT

Command Under Test: ContinuousMove, Stop

WSDL Reference: ptz.wsdl

Requirement Level: MUST IF SUPPORTED (PTZ)

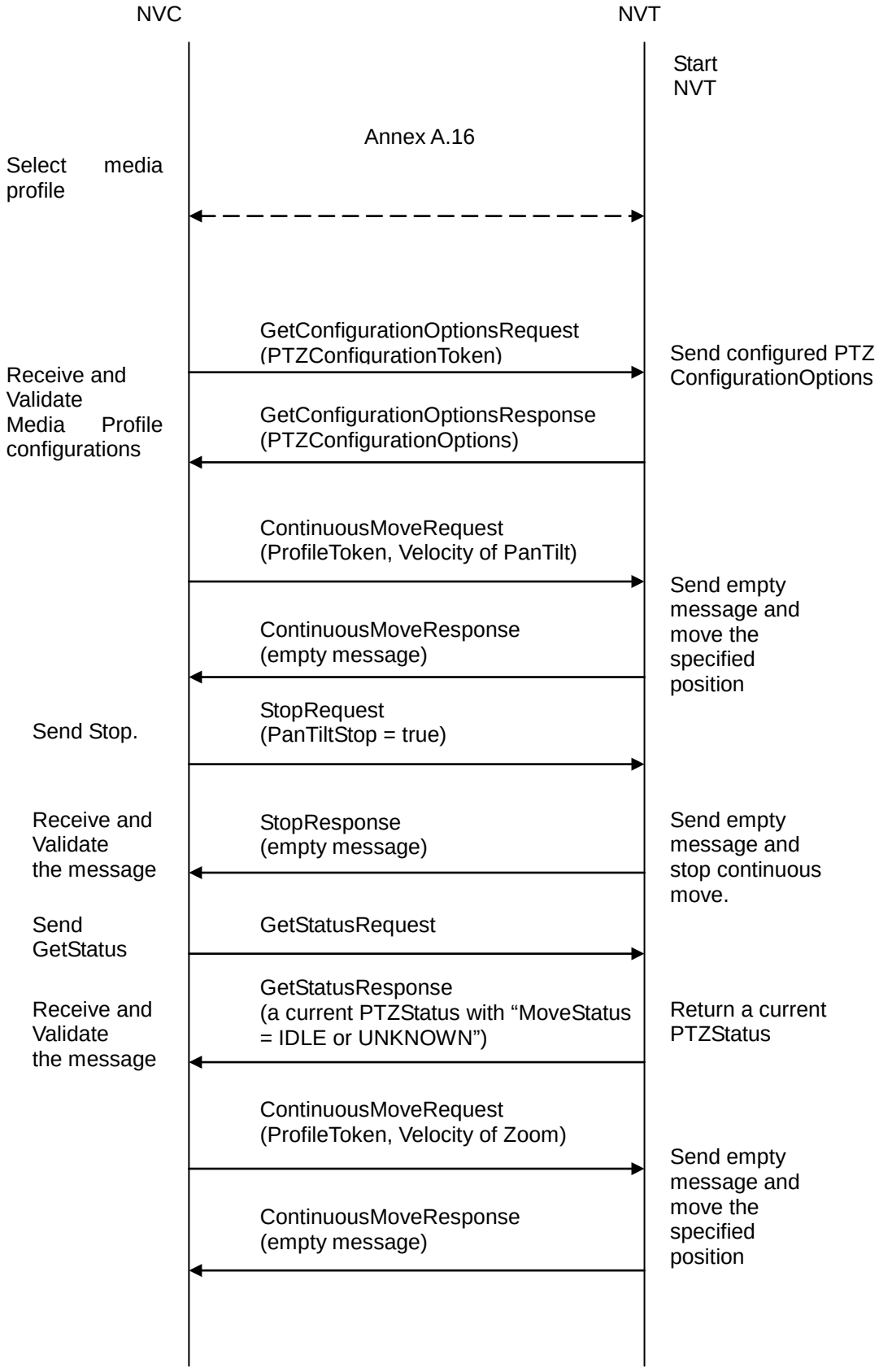
Test Purpose: To verify NVT PTZ ContinuousMove operation without timeout parameter

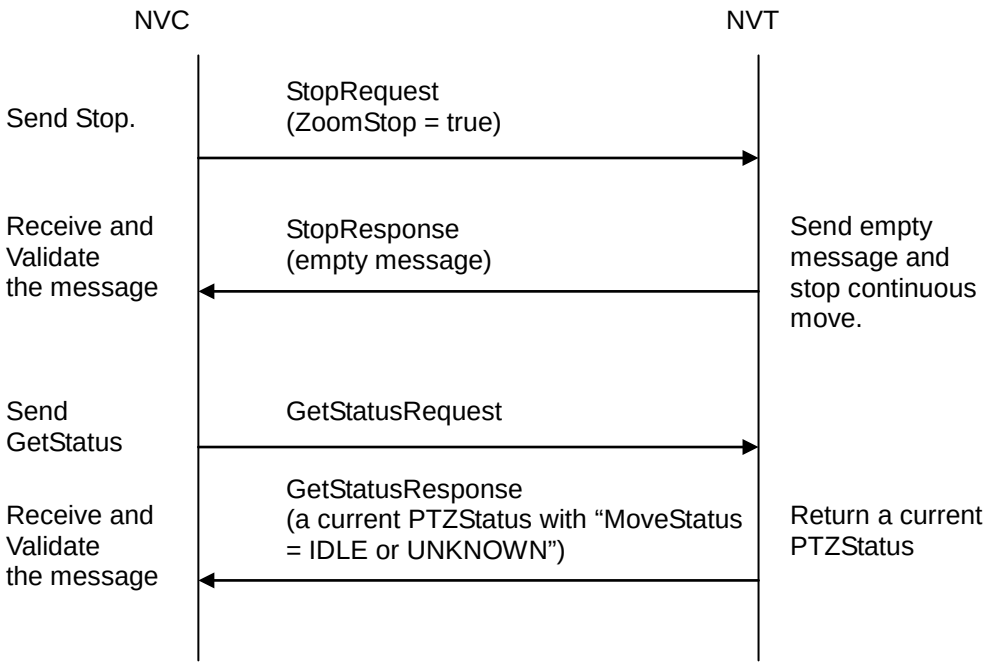
Pre-Requisite: PTZ is supported by NVT, and a function of Continuous movements is implemented. In addition, NVC gets the ptz service entry point by GetCapabilities command.

A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:





Test Procedure:

1. Start an NVC.
2. Start an NVT.
3. NVC configures and selects a media profile as described in Annex A.16.
4. NVC will invoke GetConfigurationOptionsRequest message to retrieve PTZConfigurationOptions.
5. Verify that NVT returns GetConfigurationOptionsResponse with valid Spaces and PTZTimeout, and has the function of Relative movement.
6. If PanTilt of Continuous movement is supported (there is a parameter of ContinuousPanTiltVelocitySpace in PTZConfigurationOptions), NVC will invoke ContinuousMoveRequest message (**ProfileToken, Velocity:PanTilt** = **["x", "y"]**).
7. If NVC invoked ContinuousMoveRequest message for PanTilt, verify that NVT returns ContinuousMoveResponse message indicating success.
8. If NVC invoked ContinuousMoveRequest message for PanTilt, NVC will invoke StopRequest message to stop continuous move.
9. If NVC invoked ContinuousMoveRequest message for PanTilt, verify that the NVT returns StopResponse message indicating success.
10. If NVC invoked ContinuousMoveRequest message for PanTilt, NVC will invoke GetStatusRequest message to get a current PTZStatus.
11. If NVC invoked ContinuousMoveRequest message for PanTilt, verify that the NVT returns GetStatusResponse with “MoveStatus = IDLE or UNKNOWN”.

12. If Zoom of Continuous movement is supported (there is a parameter of ContinuousZoomVelocitySpace in PTZConfigurationOptions), NVC will invoke ContinuousMoveRequest message (**ProfileToken, Velocity:Zoom** = **["x"]**).
13. If NVC invoked ContinuousMoveRequest message for Zoom, verify that NVT returns ContinuousMoveResponse message indicating success.
14. If NVC invoked ContinuousMoveRequest message for Zoom, NVC will invoke StopRequest message to stop continuous move.
15. If NVC invoked ContinuousMoveRequest message for Zoom, verify that the NVT returns StopResponse message indicating success.
16. If NVC invoked ContinuousMoveRequest message for Zoom, NVC will invoke GetStatusRequest message to get a current PTZStatus.
17. If NVC invoked ContinuousMoveRequest message for PanTilt, verify that the NVT returns GetStatusResponse with “MoveStatus = IDLE or UNKNOWN”.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT did not send GetConfigurationOptionsResponse message.

The DUT did not send valid GetConfigurationOptionsResponse message.

The DUT did not send GetConfigurationOptionsResponse message with valid Spaces and PTZTimeout.

The DUT did not send GetStatusResponse message.

The DUT did not send valid GetStatusResponse message.

The DUT did not send ContinuousMoveResponse message.

The DUT did not send StopResponse message.

The DUT did not send GetStatusResponse message with “MoveStatus = MOVING or UNKNOWN” after executing Test Procedure 9 and 17.

The DUT did not send GetStatusResponse message with “MoveStatus = IDLE or UNKNOWN” after executing Test Procedure 13 and 21.

Note: If NVT does not return a current MoveStatus by GetStatusResponse, the MoveStatus isn’t checked by NVC.

10.4 Preset operations

10.4.1 NVT SET AND GET PRESET

Test Label: PTZ NVT Set and Get Preset

ONVIF Core Specification Coverage: 13.5.1 SetPreset, 13.5.2 GetPresets.

Device Type: NVT

Command Under Test: SetPreset, GetPresets

WSDL Reference: ptz.wsdl

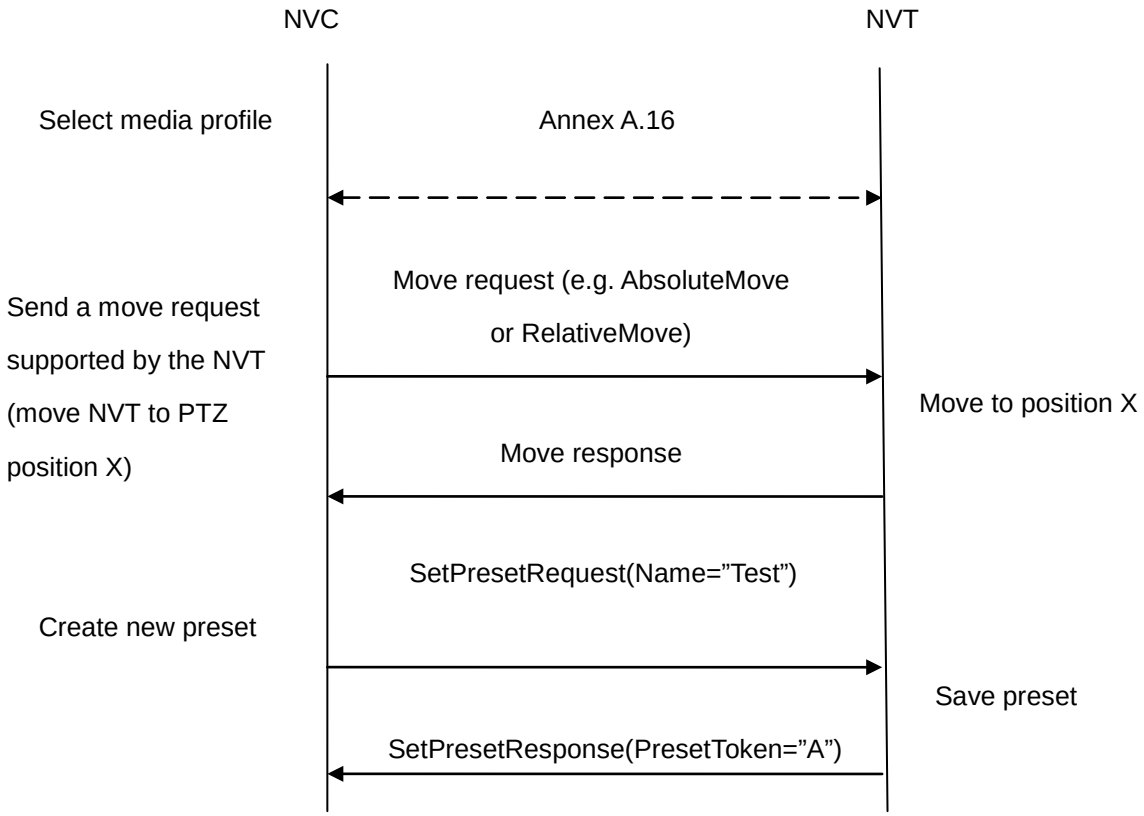
Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Presets) & IMPLEMENTED (AbsoluteMove or RelativeMove)

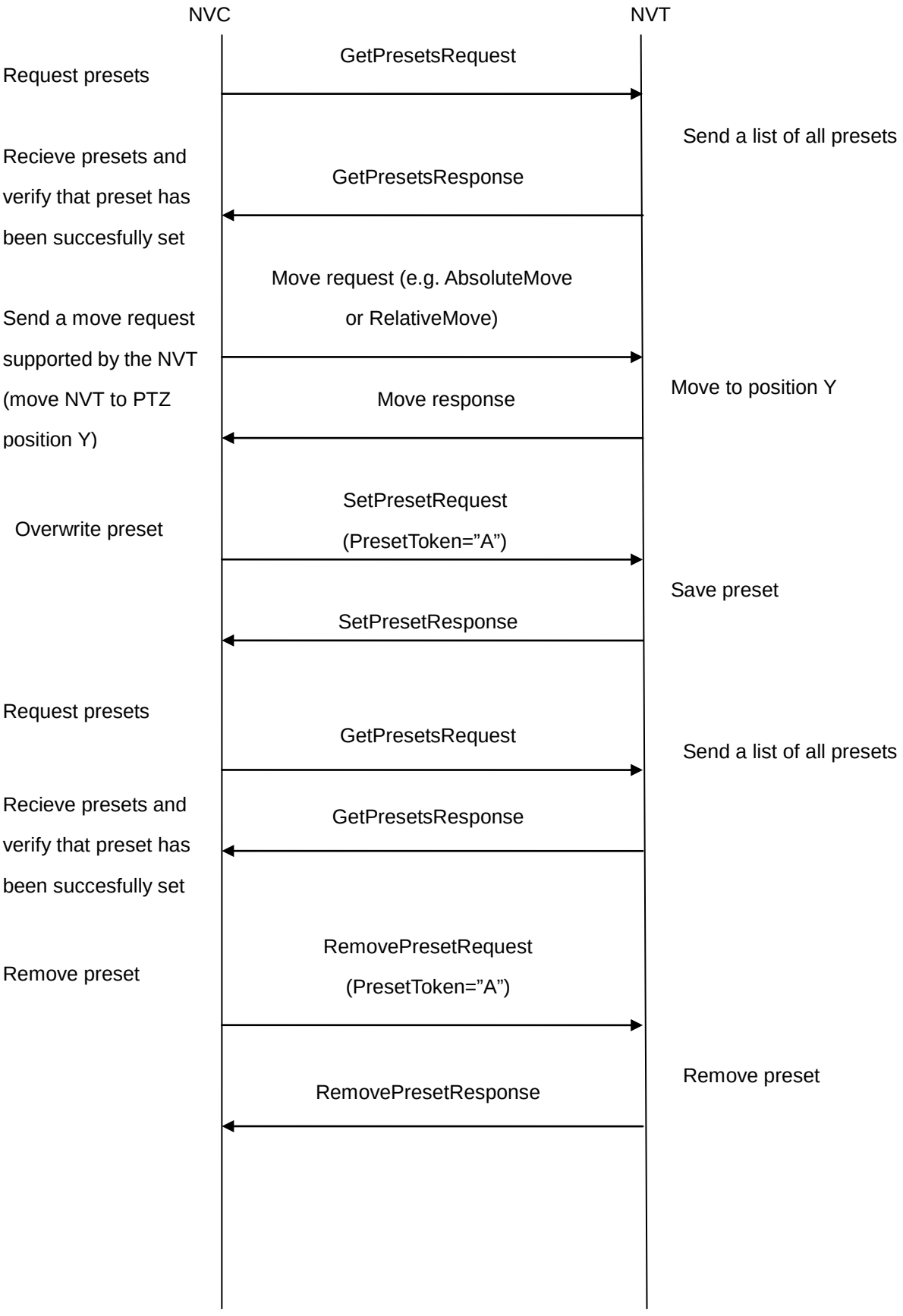
Test Purpose: To verify that the NVT supports the setting of presets using the SetPreset operation and the retrieval of presets using the GetPresets operation.

Pre-Requisite: A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Select media profile
Annex A.16
Move request (e.g. AbsoluteMove or RelativeMove)
Send a move request supported by the NVT (move NVT to PTZ position X)
Move to position X
Move response
SetPresetRequest(Name="Test")
Create new preset
Save preset
SetPresetResponse(PresetToken="A")


NVC
NVT
GetPresetsRequest
Request presets
Send a list of all presets
Recieve presets and verify that preset has been succesfully set
GetPresetsResponse
Move request (e.g. AbsoluteMove or RelativeMove)
Send a move request supported by the NVT (move NVT to PTZ position Y)
Move to position Y
Move response
SetPresetRequest (PresetToken="A")
Overwrite preset
Save preset
SetPresetResponse
Request presets
GetPresetsRequest
Send a list of all presets
Recieve presets and verify that preset has been succesfully set
GetPresetsResponse
RemovePresetRequest (PresetToken="A")
Remove preset
Remove preset
RemovePresetResponse

Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. Position the NVT so that is at PTZPosition X using a move request supported by the NVT (e.g. AbsoluteMove or RelativeMove).
3. Create a new preset using SetPresetRequest (Name="Test").
4. Verify that the NVT sends a SetPresetResponse and a PresetToken for the preset. The PresetToken will need to be used in the following test steps. The PresetToken can have any valid value but it will be referred to as "PresetToken="A" in this test case.
5. NVC sends a GetPresetsRequest.
6. NVT sends a list of presets in the GetPresetsResponse.
7. Verify that the GetPresetsResponse has a preset with PresetToken="A", Name="Test" and PTZPosition X.
8. Position the NVT so that is at PTZPosition Y using a move request supported by the NVT (e.g. AbsoluteMove or RelativeMove).
9. Overwrite the preset using SetPresetRequest (PresetToken="A").
10. NVC sends a GetPresetsRequest.
11. NVT sends a list of presets in the GetPresetsResponse.
12. Verify that there is a preset with PresetToken="A", Name="Test" and PTZPosition Y.
13. NVC sends a RemovePresetRequest (PresetToken="A") to the NVT and the NVT removes the preset.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT's move operation failed.

DUT did not send SetPresetResponse message.

DUT did not include a PresetToken in the SetPresetResponse message.

DUT did not send GetPresetsResponse message.

DUT did not include the correct PTZPosition in the GetPresetsResponse message.

DUT did not include the correct name (Name="Test") in the GetPresetsResponse message.

Note: There are no specific requirements on what the exact values for PTZPositions X and Y should be used in this test, other than they must be different positions.

PTZ accuracy is out of scope for this Test Specification. Therefore the position reported by the NVT in the GetPresetResponse does not have to be exactly the same as the position of the preset created with the SetPresetRequest.

10.4.2 NVT GOTO PRESET

Test Label: PTZ NVT GotoPreset

ONVIF Core Specification Coverage: 13.5.3 GotoPreset, 13.5.1 SetPreset

Device Type: NVT

Command Under Test: GotoPreset

WSDL Reference: ptz.wsdl

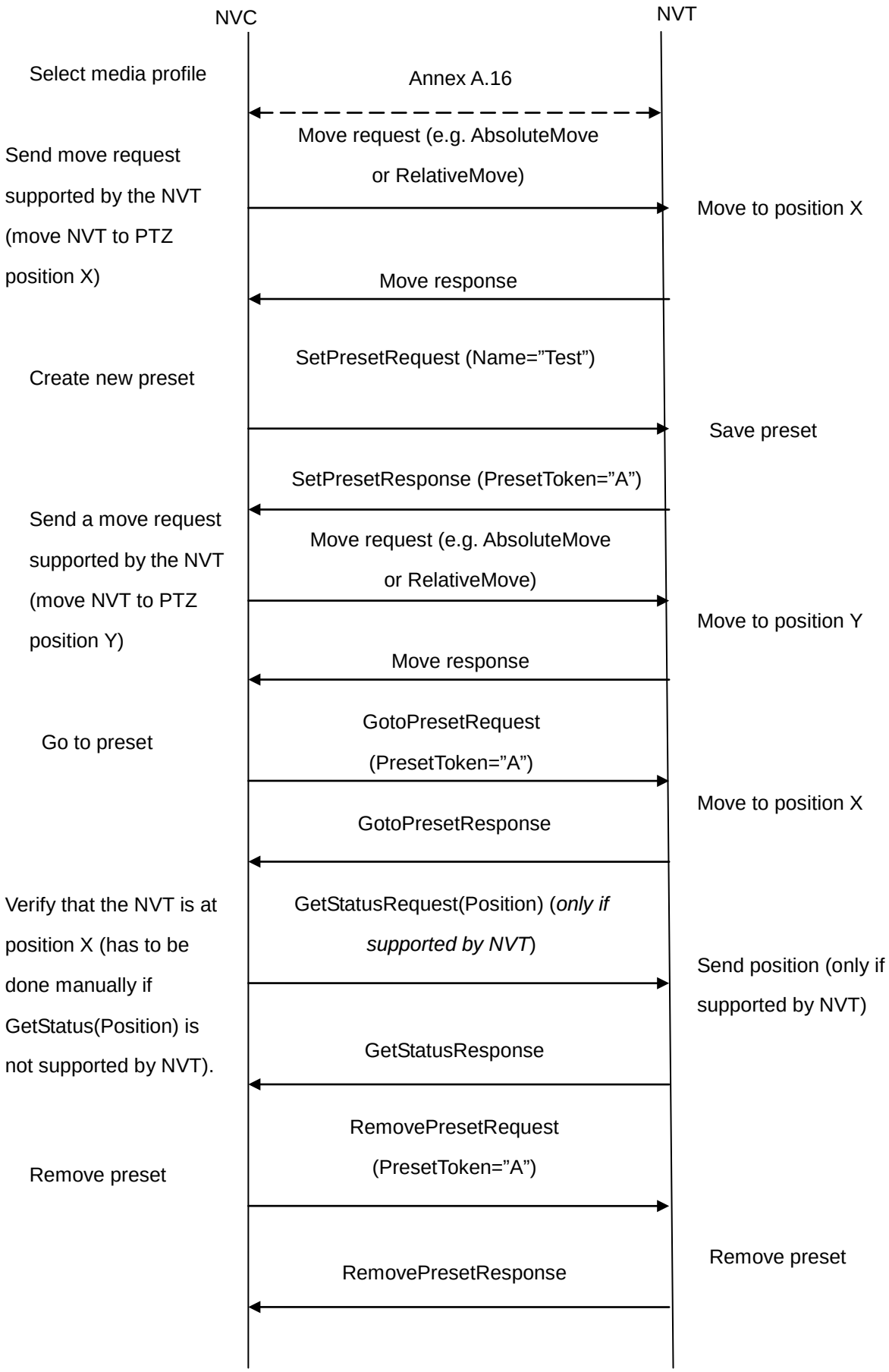
Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Presets) & IMPLEMENTED (AbsoluteMove or RelativeMove)

Test Purpose: To verify that it is possible to go to presets using the GotoPreset operation.

Pre-Requisite: A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Select media profile
Annex A.16
Move request (e.g. AbsoluteMove or RelativeMove)
Send move request supported by the NVT (move NVT to PTZ position X)
Move to position X
Move response
SetPresetRequest (Name="Test")
Create new preset
Save preset
SetPresetResponse (PresetToken="A")
Send a move request supported by the NVT (move NVT to PTZ position Y)
Move request (e.g. AbsoluteMove or RelativeMove)
Move to position Y
Move response
GotoPresetRequest (PresetToken="A")
Go to preset
Move to position X
GotoPresetResponse
Verify that the NVT is at position X (has to be done manually if GetStatus(Position) is not supported by NVT).
GetStatusRequest(Position) (*only if supported by NVT*)
Send position (only if supported by NVT)
GetStatusResponse
RemovePresetRequest (PresetToken="A")
Remove preset
Remove preset
RemovePresetResponse

Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. Position the NVT so that is at PTZPosition X using a move request supported by the NVT (e.g. AbsoluteMove or RelativeMove).
3. Create a new preset using SetPresetRequest (Name="Test").
4. Verify that the NVT sends a SetPresetResponse and a PresetToken for the preset. The PresetToken will need to be used in the following test steps. The PresetToken can have any valid value but it will be referred to as "PresetToken="A" in this test case.
5. Move NVT so that it is **not** at PTZPosition X (e.g. using AbsoluteMove Y).
6. NVC sends GotoPresetRequest (PresetToken="A").
7. NVT goes to the preset PTZ position and sends a GotoPresetResponse.
8. Verify that the NVT is at PTZPosition X. GetStatus(Position) can be used if it is supported, else this will have to be done manually.
9. NVC sends a RemovePresetRequest (PresetToken="A") to the NVT and the NVT removes the preset.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT's move operation failed.

DUT did not send SetPresetResponse message with a PresetToken.

DUT did not go to the correct position after GotoPresetRequest was sent.

DUT did not send GotoPresetResponse.

Note: There is no specific requirement on what the exact value for PTZPosition X should be used in this test case.

PTZ accuracy is out of scope for this Test Specification. Therefore the position reported by the NVT in the GetStatusResponse does not have to be exactly the same as the position of the preset created with the SetPresetRequest.

10.4.3 NVT REMOVE PRESET

Test Label: PTZ NVT RemovePreset

ONVIF Core Specification Coverage: 13.5.4 RemovePreset, 13.5.1 SetPreset, 13.5.2 GetPresets

Device Type: NVT

Command Under Test: RemovePreset

WSDL Reference: ptz.wsdl

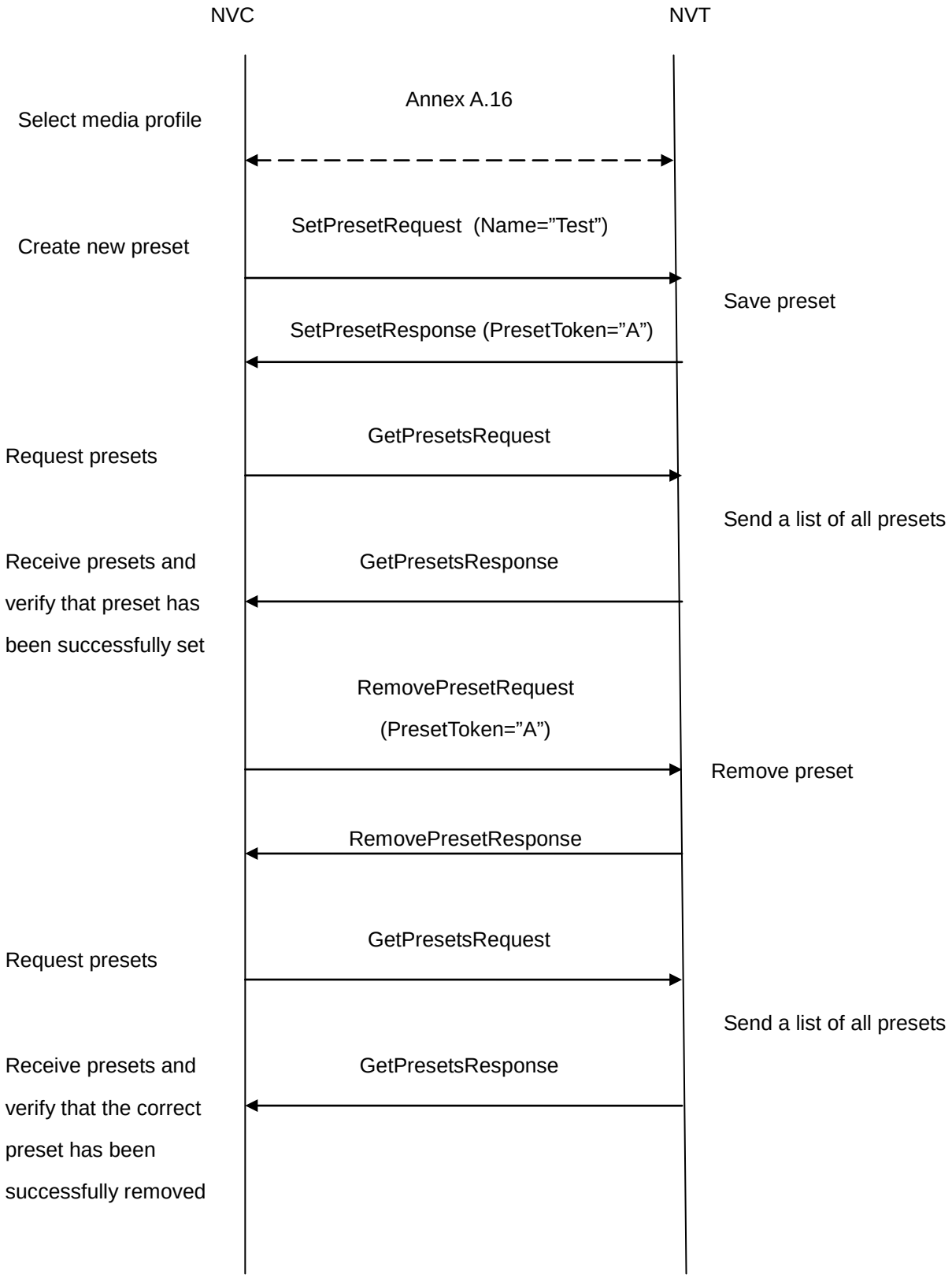
Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED Presets)

Test Purpose: To verify that it is possible to remove presets using the RemovePreset operation.

Pre-Requisite: A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Annex A.16
Select media profile
SetPresetRequest (Name="Test")
Create new preset
Save preset
SetPresetResponse (PresetToken="A")
GetPresetsRequest
Request presets
Send a list of all presets
Receive presets and verify that preset has been successfully set
GetPresetsResponse
RemovePresetRequest (PresetToken="A")
Remove preset
RemovePresetResponse
GetPresetsRequest
Request presets
Send a list of all presets
Receive presets and verify that the correct preset has been successfully removed
GetPresetsResponse

Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. NVC creates a new preset using SetPresetRequest (Name="Test")
3. NVT saves the preset and sends a SetPresetResponse. Verify that the NVT sends a SetPresetResponse and a PresetToken for the preset. The PresetToken will need to be used in the following test steps. The PresetToken can have any valid value but it will be referred to as "PresetToken="A" in this test case.
4. NVC sends a GetPresetsRequest.
5. NVT sends a list of presets in the GetPresetsResponse.
6. Verify that there is a preset with PresetToken="A" and Name="Test".
7. NVC sends RemovePresetRequest (PresetToken="A")
8. NVT removes preset and sends a RemovePresetResponse
9. NVC sends a GetPresetsRequest.
10. NVT sends a list of presets in the GetPresetsResponse.
11. Verify that there is no preset with PresetToken="A" and Name="Test".

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send SetPresetResponse message with a PresetToken.

DUT did not send GetPresetsResponse message.

DUT did not remove preset after RemovePresetRequest was sent.

DUT did not send RemovePresetResponse.

10.5 Home Position operations

10.5.1 NVT HOME POSITION OPERATIONS (CONFIGURABLE)

Test Label: PTZ NVT Configurable Home Position

ONVIF Core Specification Coverage: 13.6.1 GotoHomePosition, 13.6.2 SetHomePosition.

Device Type: NVT

Command Under Test: SetHomePosition, GotoHomePosition

WSDL Reference: ptz.wsdl

Requirement Level: MUST IF SUPPORTED (PTZ &) IMPLEMENTED (Configurable Home position) & IMPLEMENTED (AbsoluteMove or RelativeMove)

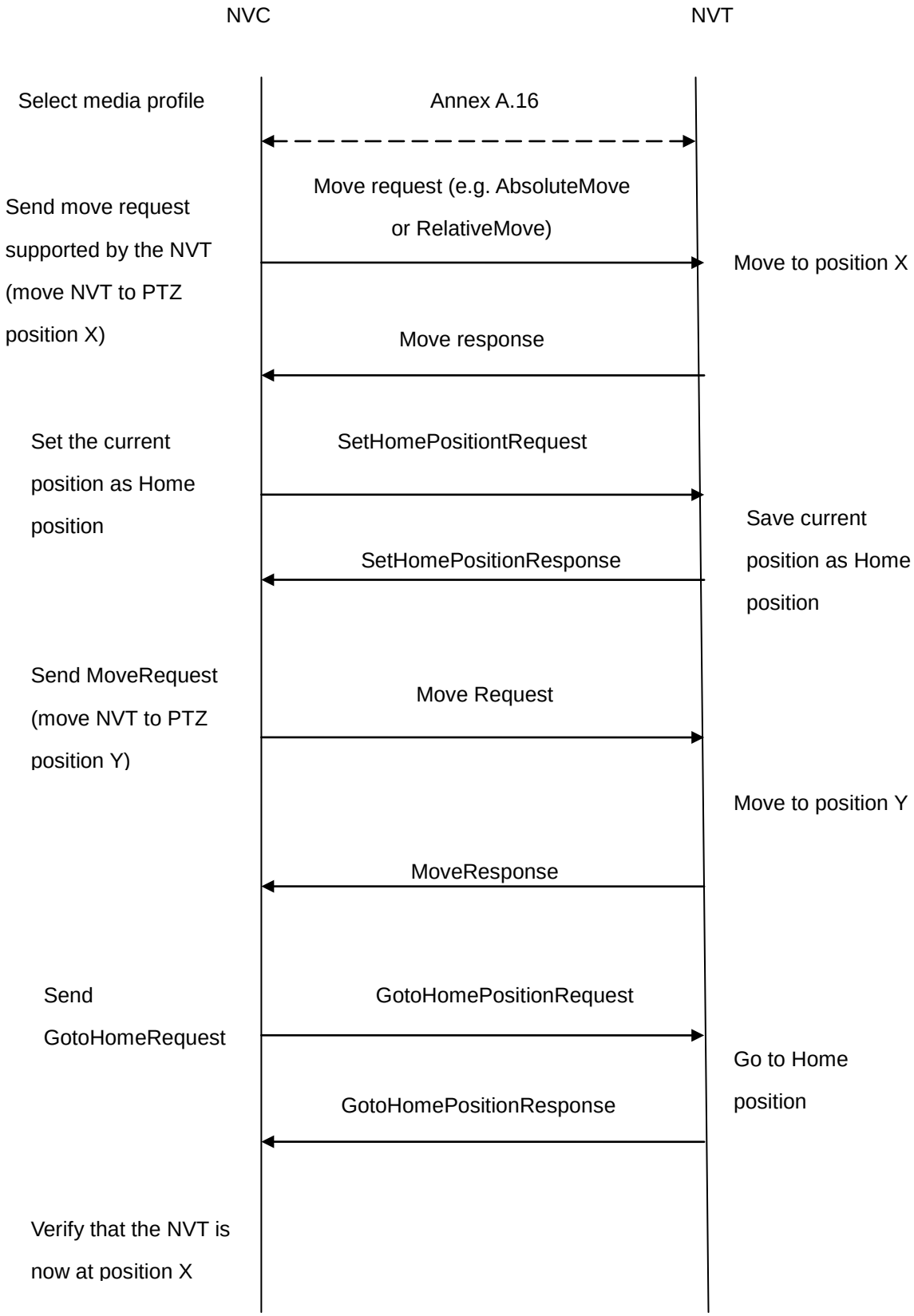
Test Purpose: To verify that the SetHomePosition and GotoHomePosition operations are correctly implemented.

Pre-Requisite:

- A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.
- This test case applies to PTZ nodes that support Configurable Home position

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Select media profile
Annex A.16
Move request (e.g. AbsoluteMove or RelativeMove)
Send move request supported by the NVT (move NVT to PTZ position X)
Move to position X
Move response
Set the current position as Home position
SetHomePositiontRequest
Save current position as Home position
SetHomePositionResponse
Send MoveRequest (move NVT to PTZ position Y)
Move Request
Move to position Y
MoveResponse
Send GotoHomeRequest
GotoHomePositionRequest
Go to Home position
GotoHomePositionResponse
Verify that the NVT is now at position X

Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. Position the NVT so that is at PTZPosition X using a move request supported by the NVT (e.g. AbsoluteMove or RelativeMove).
3. NVC sends a SetHomePositionRequest.
4. NVT sets the Home position to the current position and sends a SetHomePositionResponse.
5. Move NVT so that it is not at PTZPosition X (e.g. using AbsoluteMove Y)
6. NVC sends a GotoHomePositionRequest.
7. NVT goes to the Home PTZ position and sends a GotoHomePositionResponse.
8. Verify that the NVT is at PTZPosition X (GetStatus/Position can be used if it is supported, else this will have to be done manually)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT's move operation failed.

DUT did not send SetHomePositionResponse message.

DUT did not save the new position as Home position.

DUT did not send GotoHomePositionResponse message.

DUT did not go to Home position.

Note: PTZ accuracy is out of scope for this Test Specification. Therefore the position reported by the NVT in the GetStatusResponse does not have to be exactly the same as the position of the Home position.

10.5.2 NVT HOME POSITION OPERATIONS (FIXED)

Test Label: PTZ NVT Fixed Home Position

ONVIF Core Specification Coverage: 13.6.1 GotoHomePosition, 13.6.2 SetHomePosition.

Device Type: NVT

Command Under Test: SetHomePosition, GotoHomePosition

WSDL Reference: ptz.wsdl

Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Fixed Home position) & IMPLEMENTED (AbsoluteMove or RelativeMove)

Test Purpose: To verify that the SetHomePosition and GotoHomePosition operations are correctly implemented.

Pre-Requisite:

- A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.
- This test case applies to PTZ nodes that support fixed Home position

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Select media profile
Annex A.16
Send GotoHomeRequest
GotoHomePositionRequest
Go to Home position
GotoHomePositionResponse
Note the current PTZ position.
Send move request supported by the NVT (move NVT to PTZ position Y)
Move request (e.g. AbsoluteMove or RelativeMove)
Move to position Y
Move response
Try setting the current position as Home position
SetHomePositiontRequest
Reply with “Cannot Overwrite Home message”
SOAP Fault message (CannotOverwriteHome)
Send GotoHomeRequest
GotoHomePositionRequest
Go to Home position
GotoHomePositionResponse
Verify that the NVT is now at the same position as after the previous GotoHome request.

Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. NVC sends a GotoHomePositionRequest.
3. NVT goes to the Home position and sends a GotoHomePositionResponse.
4. Note at which PTZPosition the NVT is (GetStatus/Position can be used if it is supported, else this will have to be done manually). This position will be referred to as “PTZPosition A” below.
5. Position the NVT so that is at PTZPosition Y using a move request supported by the NVT (e.g. AbsoluteMove or RelativeMove).
6. NVC sends a SetHomePositionRequest.
7. NVT responds with “Cannot Overwrite Home” message.
8. NVC sends a GotoHomePositionRequest.
9. NVT goes to the Home PTZ position and sends a GotoHomePositionResponse.
10. Verify that the NVT is back at PTZPosition A (GetStatus/Position can be used if it is supported, else this will have to be done manually)

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not send SOAP Fault message (CannotOverwriteHome).

DUT’s move operation failed.

DUT did save the new position (“PTZPosition Y”) as Home position.

DUT did not send GotoHomePositionResponse message.

DUT did not go to original Home position (“PTZPosition A”).

Note: PTZ accuracy is out of scope for this Test Specification. Therefore the position reported by the NVT in the GetStatusResponse does not have to be exactly the same as the position of the Home position.

10.6 Auxiliary operations

10.6.1 NVT SEND AUXILIARY COMMAND

Test Label: PTZ NVT SendAuxiliaryCommand

ONVIF Core Specification Coverage: 13.7.1 SendAuxiliaryCommand.

Device Type: NVT

Command Under Test: SendAuxiliaryCommand

WSDL Reference: ptz.wsdl

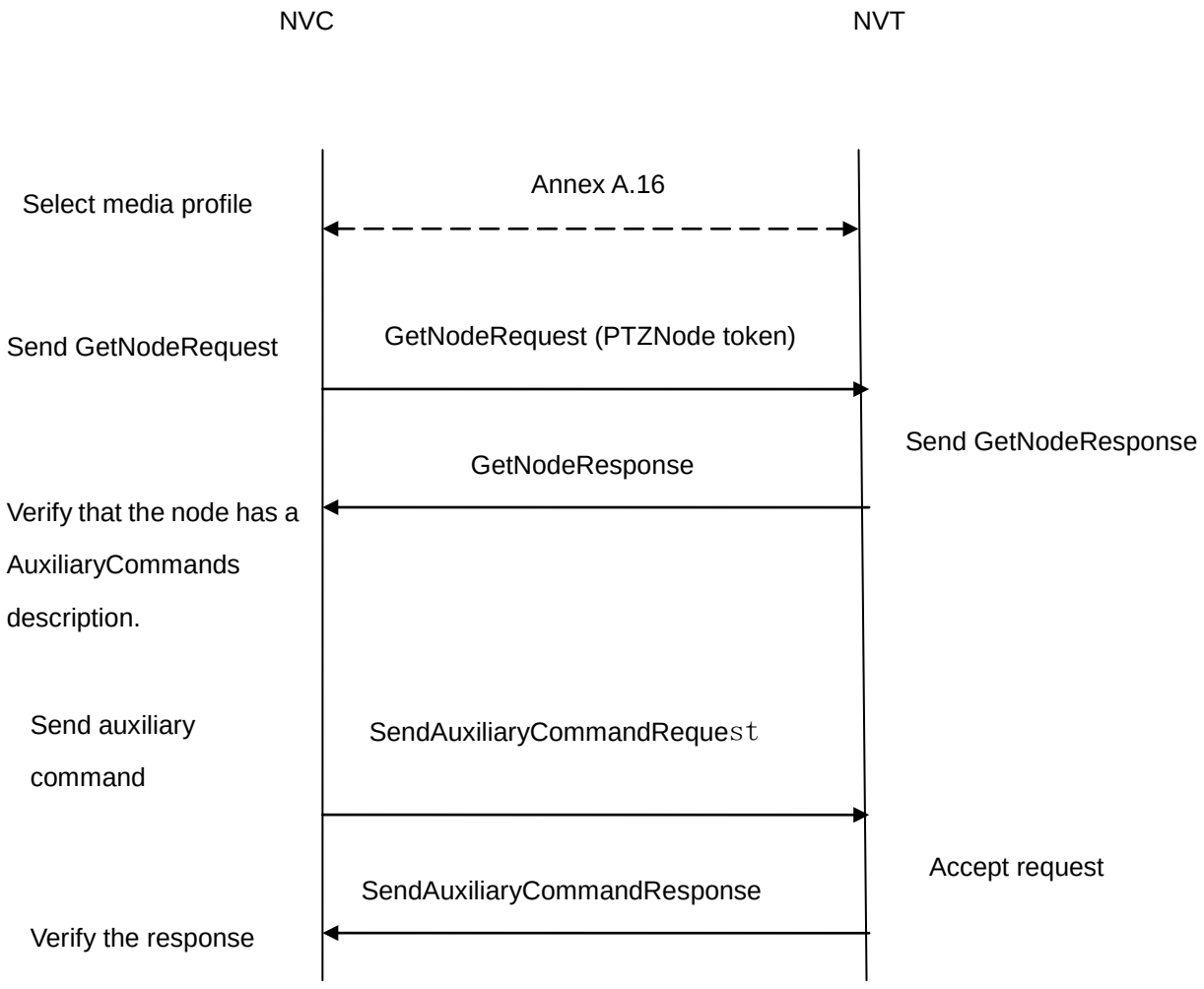
Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Auxiliary operations)

Test Purpose: To verify that it is possible to send an auxiliary command using the SendAuxiliaryCommand operation.

Pre-Requisite: A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. NVC sends a GetNodeRequest.
3. NVT sends a GetNodeResponse that includes a list of the supported auxiliary commands.
4. Send an Auxiliary command that matches the supported command listed in the PTZ Node, using SendAuxiliaryCommandRequest.
5. Verify that the NVT sends a SendAuxiliaryCommandResponse

Test Result:

PASS –

DUT passes all assertions.

FAIL –

DUT did not list the available auxiliary commands in the PTZ Node properties.

DUT did not send SendAuxiliaryCommandResponse.

Note: It is outside the scope of this test case to verify that the functionality connected to an Auxiliary command works as intended. This should be independently verified by the person executing the test.

10.7 Predefined PTZ spaces

10.7.1 Absolute Position Spaces

10.7.1.1 NVT GENERIC PAN/TILT POSITION SPACE

Test Label: PTZ NVT Absolute Position Spaces Generic Pan/Tilt

ONVIF Core Specification Coverage: 13.8.1.1 Generic Pan/Tilt Position Space

Device Type: NVT

Command Under Test: None

WSDL Reference: ptz.wsdl

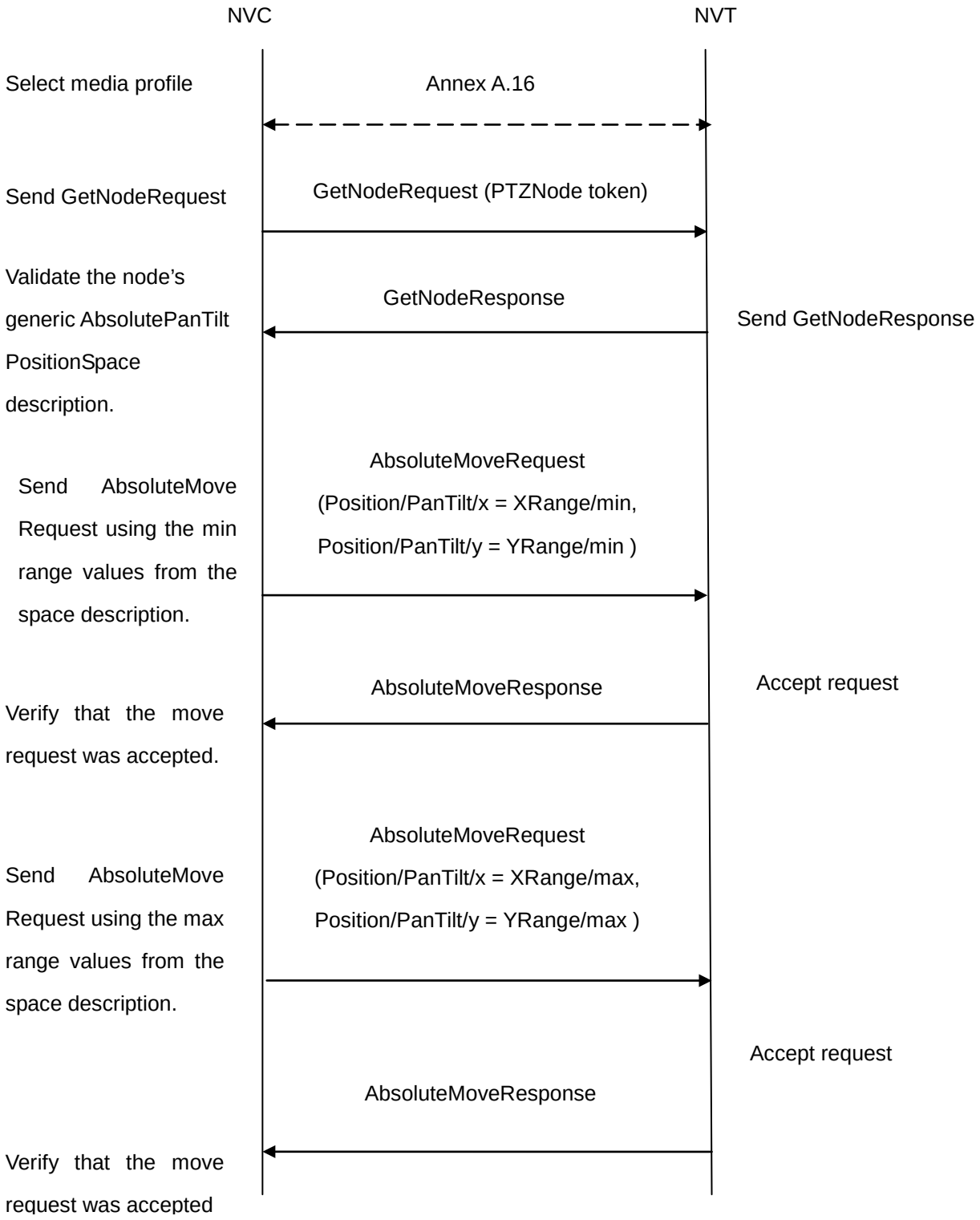
Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Absolute Move - Pan/Tilt)

Test Purpose: To verify that the node supports the Generic Pan/Tilt Position Space for AbsolutePanTilt.

Pre-Requisite: A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. NVC sends a GetNodeRequest to the NVT.
3. Verify that the node's Absolute Position Space description is correctly formed and that the allowed range is specified.
4. NVC sends a valid AbsoluteMoveRequest using the min XRange/YRange values from the space description.
5. Verify that the AbsoluteMoveRequest is accepted.
6. NVC sends a valid AbsoluteMoveRequest using the max XRange/YRange values from the space description.
7. Verify that the AbsoluteMoveRequest is accepted.
8. Repeat test procedure for all PTZNodes available in the NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT does not have a Generic Pan/Tilt Position Space description for AbsolutePanTilt.

The allowed range is not specified

A valid AbsoluteMove operation does not succeed

Note: This test case MUST be repeated for all PTZNodes available in the NVT

10.7.1.2 NVT GENERIC ZOOM POSITION SPACE

Test Label: PTZ – Absolute Position Spaces – Generic Zoom

ONVIF Core Specification Coverage: 13.8.1.2 Generic Zoom Position Space

Device Type:

NVT

Command Under Test: None

WSDL Reference: ptz.wsdl

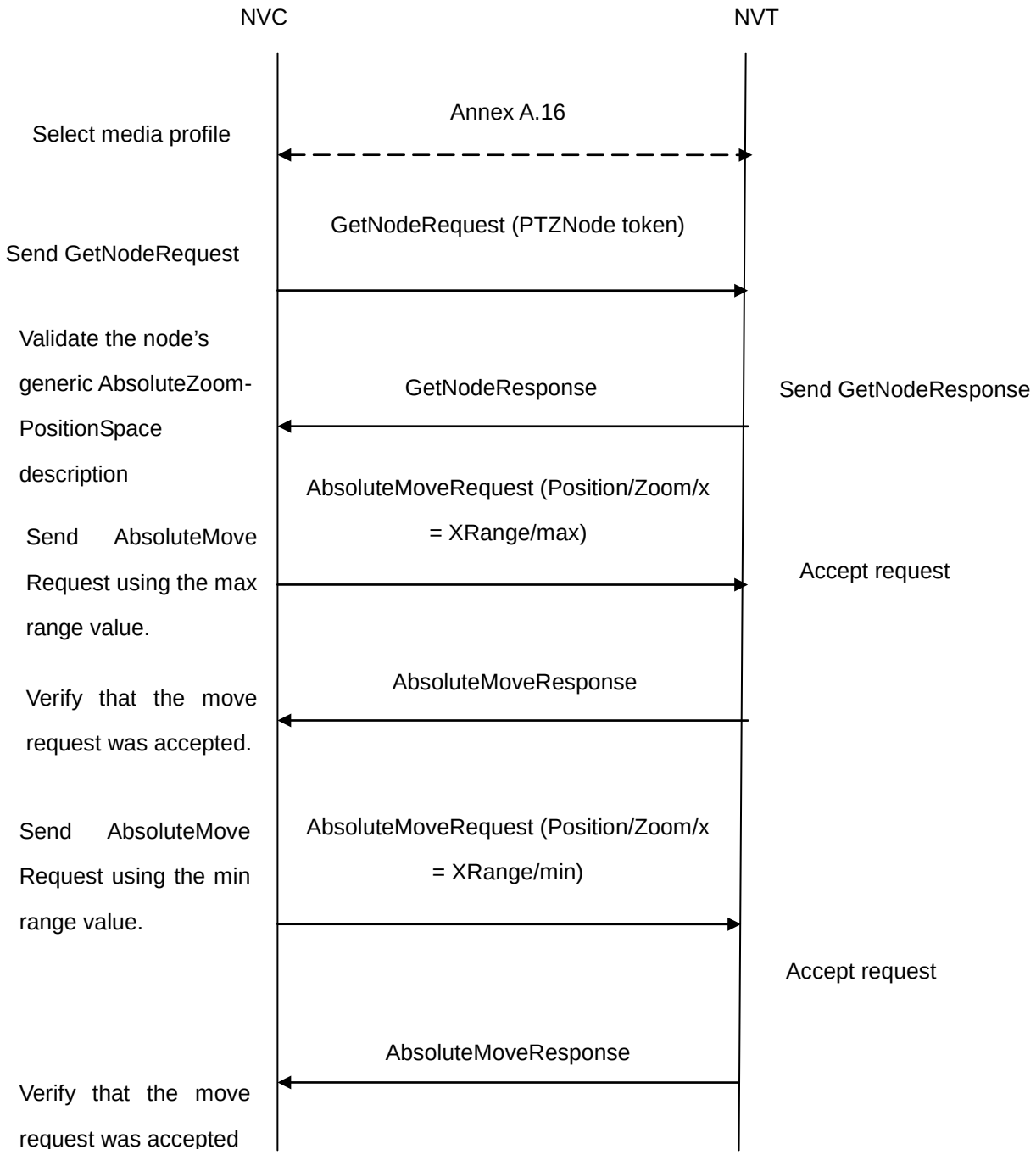
Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Absolute Move - Zoom)

Test Purpose: To verify that the node supports the Generic Zoom Position Space for Absolute Zoom.

Pre-Requisite: A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. NVC sends a GetNodeRequest to the NVT.
3. Verify that the node's Absolute Position Space description is correctly formed and that the allowed range is specified.
4. NVC sends a valid AbsoluteMoveRequest using the max Xrange value from the space description.
5. Verify that the AbsoluteMoveRequest is accepted.
6. NVC sends a valid AbsoluteMoveRequest using the min Xrange value from the space description.
7. Verify that the AbsoluteMoveRequest is accepted.
8. Repeat test procedure for all PTZNodes available in the NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT does not have a Generic Zoom Position Space description for AbsoluteZoom.

The allowed range is not specified

A valid AbsoluteMove operation does not succeed

Note: This test case MUST be repeated for all PTZNodes available in the NVT

10.7.2 Relative Translation Spaces

10.7.2.1 NVT GENERIC PAN/TILT TRANSLATION SPACE

Test Label: PTZ – Relative Translation Spaces – Generic Pan/Tilt

ONVIF Core Specification Coverage: 13.8.2.1 Generic Pan/Tilt Translation Space

Device Type:

NVT

Command Under Test: None

WSDL Reference: ptz.wsdl

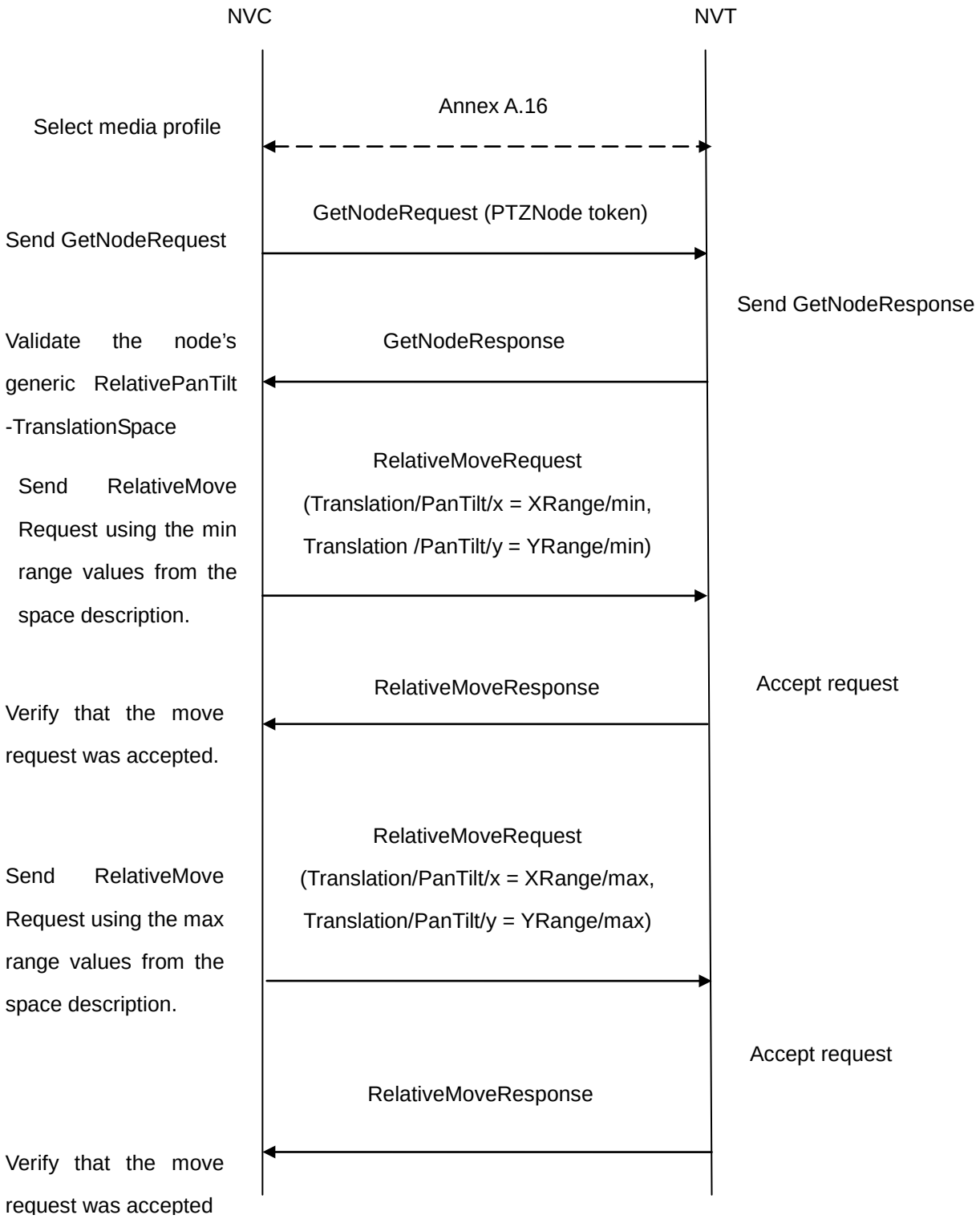
Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Relative pan/tilt)

Test Purpose: To verify that the node supports the Generic Pan/Tilt Translation Space for Relative Pan/Tilt.

Pre-Requisite: A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Annex A.16
Select media profile
GetNodeRequest (PTZNode token)
Send GetNodeRequest
Send GetNodeResponse
Validate the node's generic RelativePanTilt -TranslationSpace
GetNodeResponse
RelativeMoveRequest (Translation/PanTilt/x = XRange/min, Translation /PanTilt/y = YRange/min)
Send RelativeMove Request using the min range values from the space description.
RelativeMoveResponse
Accept request
Verify that the move request was accepted.
RelativeMoveRequest (Translation/PanTilt/x = XRange/max, Translation/PanTilt/y = YRange/max)
Send RelativeMove Request using the max range values from the space description.
Accept request
RelativeMoveResponse
Verify that the move request was accepted

Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. NVC sends a GetNodeRequest to the NVT.
3. Verify that the node's Relative Translation Space description is correctly formed and that the allowed range is specified.
4. NVC sends a valid RelativeMoveRequest using the min XRange/YRange values from the space description.
5. Verify that the RelativeMoveRequest is accepted.
6. NVC sends a valid RelativeMoveRequest using the max XRange/YRange values from the space description.
7. Verify that the RelativeMoveRequest is accepted.
8. Repeat test procedure for all PTZNodes available in the NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT does not have a Generic Pan/Tilt Translation Space description for Relative Pan/Tilt.

The allowed range is not specified

A valid RelativeMove operation does not succeed

Note: This test case MUST be repeated for all PTZNodes available in the NVT

10.7.2.2 NVT GENERIC ZOOM TRANSLATION SPACE

Test Label: PTZ – Relative Translation Spaces – Generic Zoom

ONVIF Core Specification Coverage: 13.8.2.2 Generic Zoom Translation Space

Device Type: NVT

Command Under Test: None

WSDL Reference: ptz.wsdl

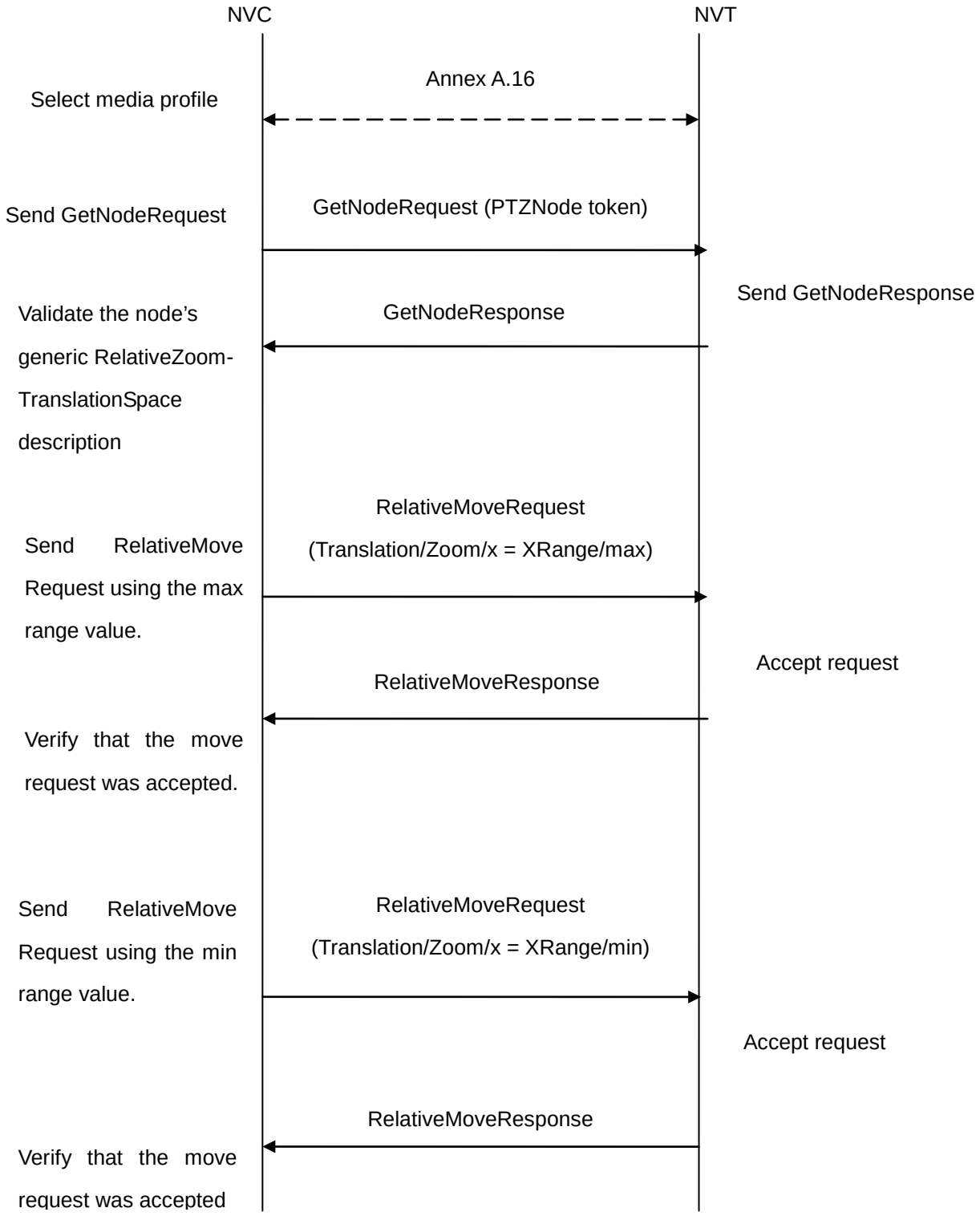
Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Relative zoom)

Test Purpose: To verify that the node supports the Generic Zoom Translation Space for Relative Zoom.

Pre-Requisite: A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. NVC sends a GetNodeRequest to the NVT.
3. Verify that the node's Relative Translation Space description is correctly formed and that the allowed range is specified.
4. NVC sends a valid RelativeMoveRequest using the max XRange value from the space description.
5. Verify that the RelativeMoveRequest is accepted.
6. NVC sends a valid RelativeMoveRequest using the min XRange value from the space description.
7. Verify that the RelativeMoveRequest is accepted.
8. Repeat test procedure for all PTZNodes available in the NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT does not have a Generic Zoom Translation Space description for RelativeZoom.

The allowed range is not specified

A valid RelativeMove operation does not succeed

Note: This test case MUST be repeated for all PTZNodes available in the NVT

10.7.3 Continuous Velocity Spaces

10.7.3.1 NVT GENERIC PAN/TILT VELOCITY SPACE

Test Label: PTZ – Continuous Velocity Spaces – Generic Pan/Tilt

ONVIF Core Specification Coverage: 13.8.3.1 Generic Pan/Tilt Velocity Space

Device Type: NVT

Command Under Test: None

WSDL Reference: ptz.wsdl

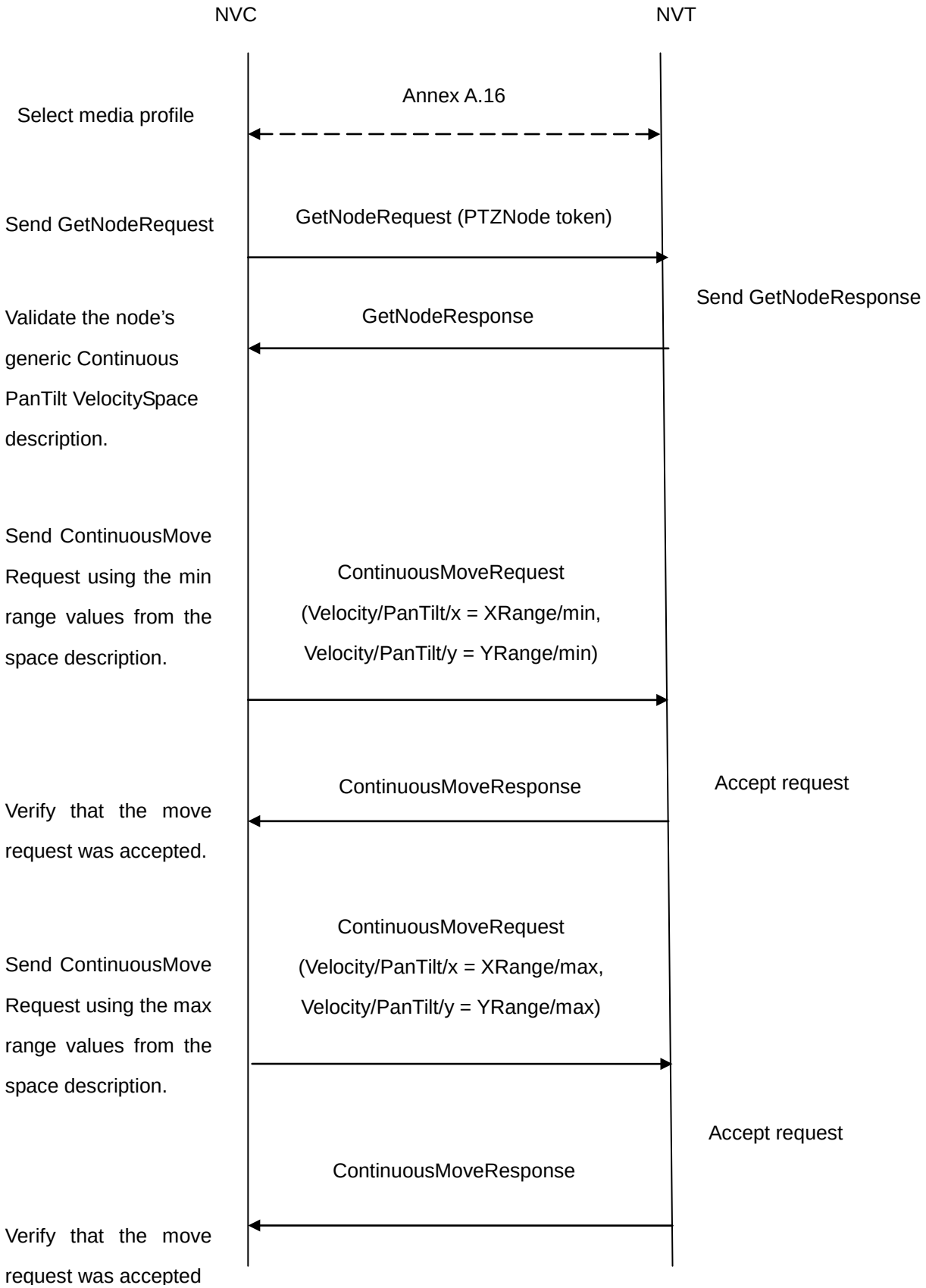
Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Pan/Tilt movement)

Test Purpose: To verify that the node supports the Generic Pan/Tilt Velocity Space for Continuous Pan/Tilt.

Pre-Requisite: A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Select media profile
Annex A.16
Send GetNodeRequest
GetNodeRequest (PTZNode token)
Send GetNodeResponse
Validate the node's generic Continuous PanTilt VelocitySpace description.
GetNodeResponse
Send ContinuousMove Request using the min range values from the space description.
ContinuousMoveRequest (Velocity/PanTilt/x = XRange/min, Velocity/PanTilt/y = YRange/min)
Accept request
Verify that the move request was accepted.
ContinuousMoveResponse
Send ContinuousMove Request using the max range values from the space description.
ContinuousMoveRequest (Velocity/PanTilt/x = XRange/max, Velocity/PanTilt/y = YRange/max)
Accept request
Verify that the move request was accepted
ContinuousMoveResponse

Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. NVC sends a GetNodeRequest to the NVT.
3. Verify that the node's Continuous Velocity Space description is correctly formed and that the allowed range is specified.
4. NVC sends a valid ContinuousMoveRequest using the min XRange/YRange values from the space description.
5. Verify that the ContinuousMoveRequest is accepted.
6. NVC sends a valid ContinuousMoveRequest using the min XRange/YRange values from the space description.
7. Verify that the ContinuousMoveRequest is accepted.
8. Repeat test procedure for all PTZNodes available in the NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT does not have a Generic Pan/Tilt Velocity Space description for Continuous Pan/Tilt.

The allowed range is not specified

A valid ContinuousMove operation does not succeed

Note: This test case MUST be repeated for all PTZNodes available in the NVT

10.7.3.2 NVT GENERIC ZOOM VELOCITY SPACE

Test Label: PTZ – Continuous Velocity Spaces – Generic Zoom

ONVIF Core Specification Coverage: 13.8.3.2 Generic Zoom Velocity Space

Device Type: NVT

Command Under Test: None

WSDL Reference: ptz.wsdl

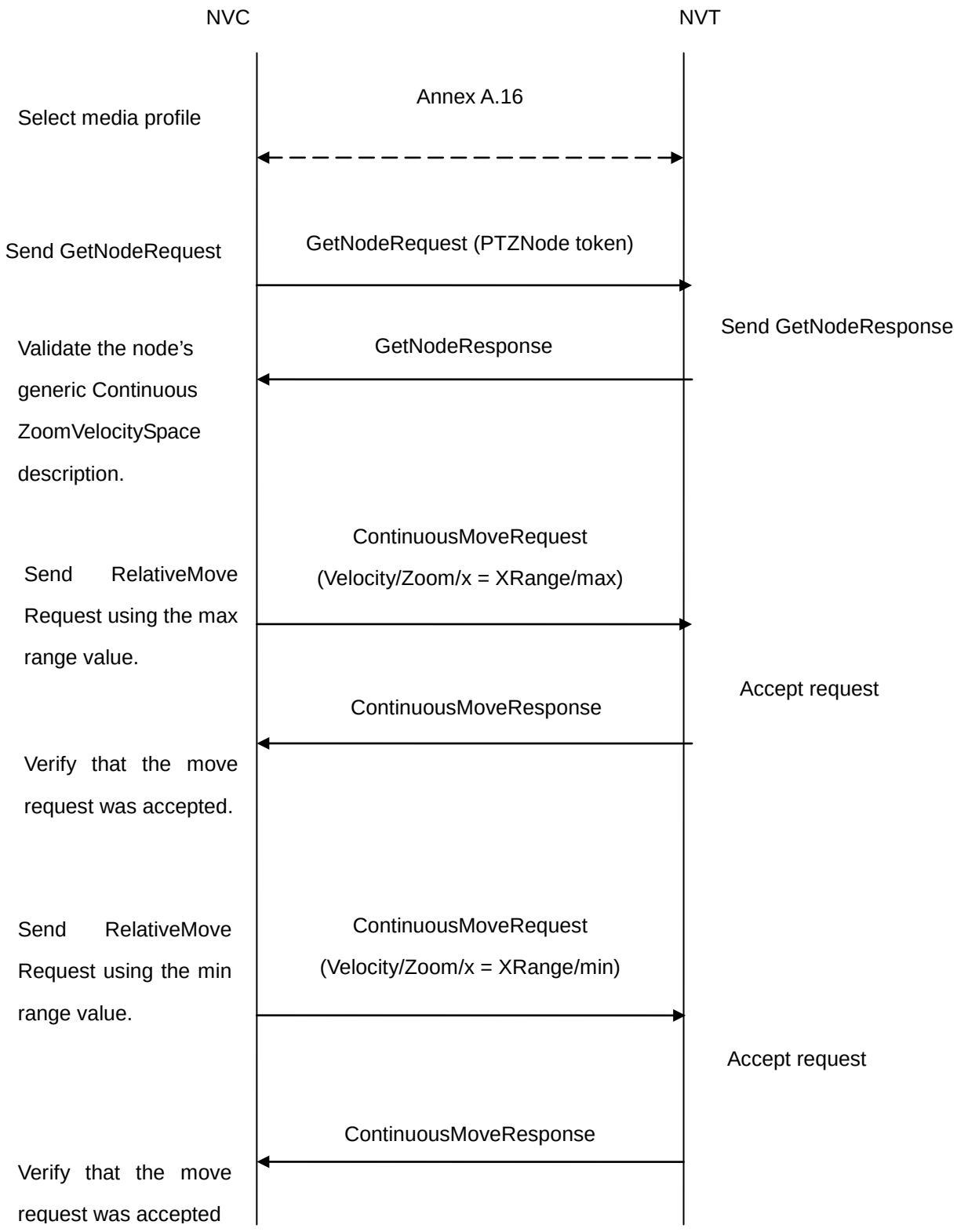
Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Zoom movement)

Test Purpose: To verify that the node supports the Generic Zoom Velocity Space for Continuous Zoom.

Pre-Requisite: A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:



Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. NVC sends a GetNodeRequest to the NVT.
3. Verify that the node's Continuous Velocity Space description is correctly formed and that the allowed range is specified.
4. NVC sends a valid ContinuousMoveRequest using the max XRange value from the space description.
5. Verify that the ContinuousMoveRequest is accepted.
6. NVC sends a valid ContinuousMoveRequest using the min XRange value from the space description.
7. Verify that the ContinuousMoveRequest is accepted.
8. Repeat test procedure for all PTZNodes available in the NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT does not have a Generic Zoom Velocity Space description for ContinuousZoom.

The allowed range is not specified

A valid ContinuousMove operation does not succeed

Note: This test case MUST be repeated for all PTZNodes available in the NVT

10.7.4 Speed Spaces

10.7.4.1 NVT GENERIC PAN/TILT SPEED SPACE

Test Label: PTZ – Speed Spaces – Generic Pan/Tilt

ONVIF Core Specification Coverage: 13.8.4.1 Generic Pan/Tilt Speed Space

Device Type: NVT

Command Under Test: None

WSDL Reference: ptz.wsdl

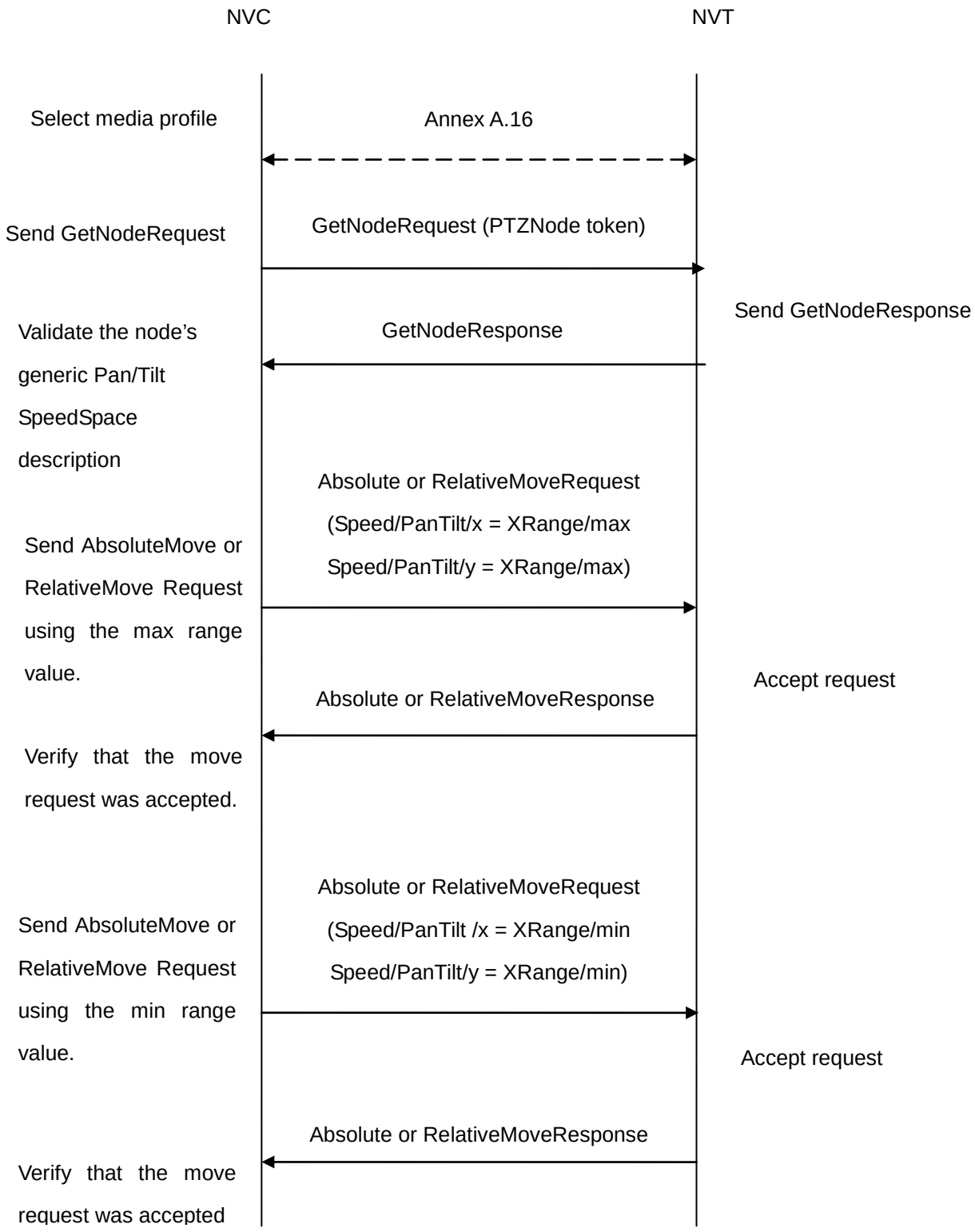
Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Speed for pan/tilt)

Test Purpose: To verify that the node supports the Generic Pan/Tilt Speed Space for pan/tilt.

Pre-Requisite: A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Select media profile
Annex A.16
Send GetNodeRequest
GetNodeRequest (PTZNode token)
Send GetNodeResponse
Validate the node's generic Pan/Tilt SpeedSpace description
GetNodeResponse
Absolute or RelativeMoveRequest (Speed/PanTilt/x = XRange/max Speed/PanTilt/y = XRange/max)
Send AbsoluteMove or RelativeMove Request using the max range value.
Accept request
Absolute or RelativeMoveResponse
Verify that the move request was accepted.
Absolute or RelativeMoveRequest (Speed/PanTilt /x = XRange/min Speed/PanTilt/y = XRange/min)
Send AbsoluteMove or RelativeMove Request using the min range value.
Accept request
Absolute or RelativeMoveResponse
Verify that the move request was accepted

Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. NVC sends a GetNodeRequest to the NVT.
3. Verify that the node's Speed Space description is correctly formed and that the allowed range is specified.
4. NVC sends a valid AbsoluteMove or RelativeMove Request (depending on which is supported by the PTZNode) using the max XRange value from the space description.
5. Verify that the AbsoluteMove (or RelativeMove) Request is accepted.
6. NVC sends a valid AbsoluteMove or RelativeMove Request (depending on which is supported by the PTZNode) using the min XRange value from the space description.
7. Verify that the AbsoluteMove (or RelativeMove) Request is accepted.
8. Repeat test procedure for all PTZNodes available in the NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT does not have a Generic Pan/Tilt Position Space description for Speed Pan/Tilt.

The allowed range is not specified

A valid AbsoluteMove or RelativeMove Request (depending on which is supported by the PTZNode) does not succeed.

Note: This test case MUST be repeated for all PTZNodes available in the NVT

10.7.4.2 NVT GENERIC ZOOM SPEED SPACE

Test Label: PTZ – Speed Spaces – Generic Zoom

ONVIF Core Specification Coverage: 13.8.4.2 Generic Zoom Speed Space

Device Type: NVT

Command Under Test: None

WSDL Reference: ptz.wsdl

Requirement Level: MUST IF SUPPORTED (PTZ) & IMPLEMENTED (Speed for zoom)

Test Purpose: To verify that the node supports the Generic Zoom Speed Space for zoom.

Pre-Requisite: A ProfileToken that refers to a Media Profile that includes a PTZConfiguration for the PTZNode is required.

Test Configuration: NVC and NVT

Test Sequence:


NVC
NVT
Select media profile
Annex A.16
Send GetNodeRequest
GetNodeRequest (PTZNode token)
Send GetNodeResponse
Validate the node's generic ZoomSpeedSpace description
GetNodeResponse
Send AbsoluteMove or RelativeMove Request using the max range value.
Absolute or RelativeMoveRequest (Speed/Zoom/x = XRange/max)
Accept request
Absolute or RelativeMoveResponse
Verify that the move request was accepted.
Send AbsoluteMove or RelativeMove Request using the min range value.
Absolute or RelativeMoveRequest (Speed/Zoom/x = XRange/min)
Accept request
Absolute or RelativeMoveResponse
Verify that the move request was accepted

Test Procedure:

1. NVC configures and selects a media profile as described in Annex A.16.
2. NVC sends a GetNodeRequest to the NVT.
3. NVT sends a GetNodeResponse
4. Verify that the node's Speed Space description is correctly formed and that the allowed range is specified.
5. NVC sends a valid AbsoluteMove or RelativeMove Request (depending on which is supported by the PTZNode) using the max XRange value from the space description.
6. Verify that the AbsoluteMove (or RelativeMove) Request is accepted.
7. NVC sends a valid AbsoluteMove or RelativeMove Request (depending on which is supported by the PTZNode) using the min XRange value from the space description.
8. Verify that the AbsoluteMove (or RelativeMove) Request is accepted.
9. Repeat test procedure for all PTZNodes available in the NVT.

Test Result:

PASS –

DUT passes all assertions.

FAIL –

The DUT does not have a Generic Pan/Tilt Speed Space description for SpeedZoom.

The allowed range is not specified

Note: This test case MUST be repeated for all PTZNodes available in the NVT

11 Security Test Cases

11.1 NVT USER TOKEN PROFILE

Test Label: Security – User token profile

ONVIF Core Specification Coverage: 5.12.2 Message level security

Device Type: NVT

Command Under Test: None

WSDL Reference: None

Requirement Level: MUST

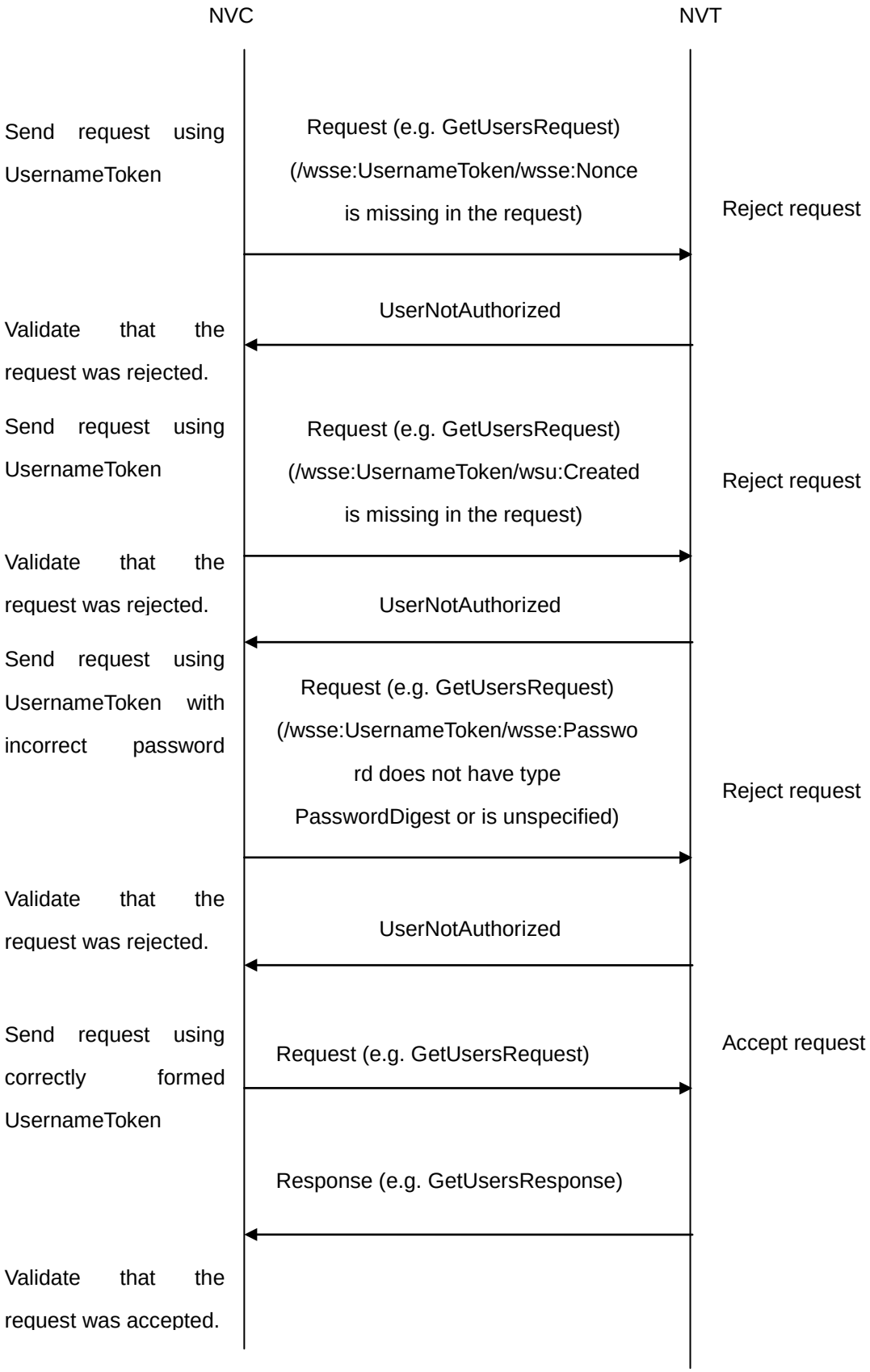
Test Purpose: To verify that the NVT supports the User Token Profile for Message level security.

Pre-Requisite:

- A user with Administrator rights (and password set) is needed in this test case. If such a user does not exist then one must be created before the test is executed. Similarly, if there exists such a user but the password has not been set, then the password must be set before the test is executed. If any such changes to user settings are needed for this test case, then they should be done before starting the test sequence and the NVT should be reset to its original settings when the test sequence is finished.
- At least one operation that requires authentication is needed in this test case. If the example given in this test case (GetUsers) does not require authentication, then the test operator must choose another operation that does require authentication.

Test Configuration: NVT and NVC

Test Sequence:


NVC
NVT
Send request using UsernameToken
Request (e.g. GetUsersRequest) (/wsse:UsernameToken/wsse:Nonce is missing in the request)
Reject request
UserNotAuthorized
Validate that the request was rejected.
Send request using UsernameToken
Request (e.g. GetUsersRequest) (/wsse:UsernameToken/wsu:Created is missing in the request)
Reject request
Validate that the request was rejected.
UserNotAuthorized
Send request using UsernameToken with incorrect password
Request (e.g. GetUsersRequest) (/wsse:UsernameToken/wsse:Password does not have type PasswordDigest or is unspecified)
Reject request
Validate that the request was rejected.
UserNotAuthorized
Send request using correctly formed UsernameToken
Request (e.g. GetUsersRequest)
Accept request
Response (e.g. GetUsersResponse)
Validate that the request was accepted.

Test Procedure:

1. NVC sends a request that requires authentication (e.g. GetUsers) to the NVT with incorrectly implemented UsernameToken. Each of the following must be tested:
    a. Missing nonce.
    b. Missing timestamp.
    c. Incorrect password type.
2. Verify that the NVT rejects all incorrect requests.
3. NVC sends a request (e.g. GetUsers) to the NVT with correctly formatted UsernameToken.
4. Verify that the NVT accepts the correct request.

Test Result

PASS –

DUT passes all assertions.

FAIL –

DUT does not support Username Token profile.

DUT accepts Username Token without nonce.

DUT accepts Username Token without timestamp.

DUT accepts Username Token without password type “PasswordDigest”.

DUT rejects Username Token with nonce and timestamp.

Note: Below is an example of a correctly formed UsernameToken for message level security, with nonce, timestamp and correct password type. For further details, refer to [ONVIF Core] and [WS-Security].

```
<SOAP:Envelope xmlns:SOAP="..." xmlns:wsse="..." xmlns:wsu= "...">
    <SOAP:Header>
        ...
        <wsse:Security>
            <wsse:UsernameToken>
                <wsse:Username>...</wsse:Username>
                <wsse:Password Type="...#PasswordDigest">...</wsse:Password>
                <wsse:Nonce>...</wsse:Nonce>
```

```
                    <wsu:Created>...</wsu:Created>
                </wsse:UsernameToken>
            </wsse:Security>
            ...
        </SOAP:Header>
        ...
</SOAP:Envelope>
```

12 Annex

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

A.1 Invalid Device Type and Scope Type

Device Type in the <d:Types:> declaration: dn:NetworkVideoTransmitter

<d:Types> list must include “NetworkVideoTransmitter” , otherwise it is considered as invalid device type.

Invalid Scope Type:

- Scope URI is not formed according to the rules of RFC 3986.

A.2 Invalid Hostname, DNSname

A string which is not formed according to the rules of RFC 952 and RFC 1123 is considered as invalid string.

A.3 Invalid Media Profile

Media profile token is a string of max length value of 64.

Invalid Media Profile:

- A string which is not formed according to the rules of RFC 952.
- A string which exceeds max length value of 64.

A.4 Invalid TimeZone

The Time Zone format is specified by POSIX, refer to POSIX 1003.1 section 8.3.

Example: Europe, Paris TZ=CET-1CEST,M3.5.0/2,M10.5.0/3

CET = designation for standard time when daylight saving is not in force.

-1 = offset in hours = negative so 1 hour east of Greenwich meridian.

CEST = designation when daylight saving is in force ("Central European Summer Time")

, = no offset number between code and comma, so default to one hour ahead for daylight saving

M3.5.0 = when daylight saving starts = the last Sunday in March (the "5th" week means the last in the month)

/2, = the local time when the switch occurs = 2 a.m. in this case

M10.5.0 = when daylight saving ends = the last Sunday in October.

/3, = the local time when the switch occurs = 3 a.m. in this case

A TimeZone token which is not formed according to the rules of POSIX 1003.1 section 8.3 is considered as invalid timezone.

A.5 Invalid RTP Header

A RTP header which is not formed according to the header field format defined in the RFC 3550 Section 5.1 is considered as invalid RTP header.

A.6 Invalid SOAP 1.2 Fault Message

A SOAP 1.2 fault message which is not formed according to the rules defined in SOAP 1.2, Part 1 Section 5.4 is considered as invalid.

A.7 Usage of URI Life Time

GetStreamUriResponse message contains the Uri to be used for requesting the media stream as well as parameters defining the lifetime of the Uri.

Valid combinations (definition of the lifetime of the Uri):

- ValidUntilConnect = true, ValidUntilReboot = false, Timeout is zero
- ValidUntilConnect = true, ValidUntilReboot = false, Timeout is non-zero
- ValidUntilConnect = false, ValidUntilReboot = true, Timeout is zero
- ValidUntilConnect = false, ValidUntilReboot = false, Timeout is non-zero

GetStreamUriResponse message which does not include any of the above valid combination is considered as invalid.

A.8 Invalid WSDL URL

An URL which is not formed according to the rules of RFC 3986 is considered as invalid WSDL URL.

A.9 Valid/Invalid IPv4 Address

IPv4 Address token is represented in dotted decimal notation (32 bit internet address is divided into four 8 bit fields and each field is represented in decimal number separated by a dot).

- Valid IPv4 addresses are in the range 0.0.0.0 to 255.255.255.255 excluding 0/8, 255/8, and 127/8, as defined in RFC 758, and 169.254/16 as defined in RFC 3927.

- Valid IPv4 addresses for a device must be valid according to the defined network mask and gateway (the gateway must be reachable and must not be identical to the assigned IPv4 address).
- Reserved addresses such as 240.0.0.0 through 255.255.255.254, as defined in RFC 2780 are prohibited for IPv4 devices.

A.10 StreamSetup syntax for GetStreamUri

The following media stream setups for GetStreamUri are covered in this Test Specification:

1. RTP unicast over UDP
2. RTP over RTSP over HTTP over TCP
3. RTP over TCP
4. RTP over RTSP over TCP

The correct syntax for the StreamSetup element for these media stream setups are as follows:

1. RTP unicast over UDP

```
<StreamSetup>
    <StreamType>RTP_unicast</StreamType>
    <Transport>
        <Protocol>UDP</Protocol>
    </Transport>
</StreamSetup>
```

2. RTP over RTSP over HTTP over TCP

```
<StreamSetup>
    <StreamType>RTP_unicast</StreamType>
    <Transport>
        <Protocol>HTTP</Protocol>
    </Transport>
</StreamSetup>
```

3. RTP over RTSP over TCP

```
<StreamSetup>
    <StreamType>RTP_unicast</StreamType>
    <Transport>
        <Protocol>RTSP</Protocol>
```

```
    </Transport>
</StreamSetup>
```

A.11 I-frame insertion time interval

'I-frame insertion time interval' is the time interval between two consecutive I-frames sent by NVT.

NVC calculates this value by using 'GovLength' parameter in the Video encoder configuration. NVC has to configure 'GovLength' to larger value so that there will be sufficient time difference between two I-frames.

In case of SetSynchronizationPoint test cases in "Real Time Streaming" section, NVC follows the following procedure to verify that I-frame is inserted as a result of "SetSynchronizationPointRequest".

1. NVC waits for an I-frame before invoking SetSynchronizationPointRequest.
2. After receiving I-frame, NVC starts a timer with time out period less than 'I-frame insertion time interval' and immediately invokes SetSynchronizationPointRequest.
3. NVC waits for the I-frame and verifies that it receives I-frame before the timeout period.

A.12 WS-Discovery timeout value

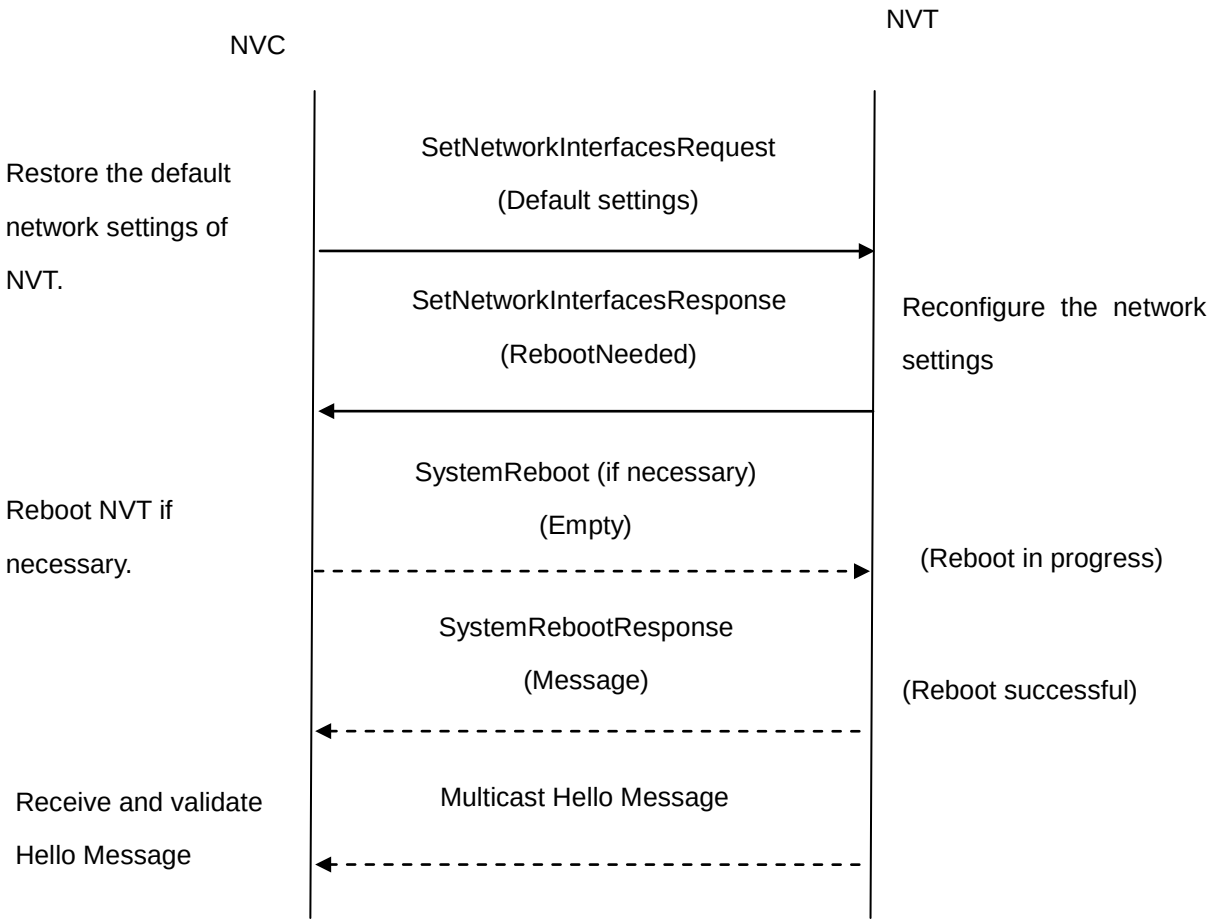
NVC uses the following timeout value in the respective Test result steps.

APP_MAX_DELAY 5 sec

A.13 Restore Network Settings

When the default network settings of the NVT are changed during the execution of Network configuration related test cases, NVC follows the following procedure to restore the original default settings at the end of actual test sequence.

1. Restore the default network settings by invoking SetNetworkInterfaces (**Default settings**) command.
2. If Reboot is needed by NVT, invoke SystemReboot command.
3. If Systemreboot is invoked, wait for HELLO message from the default network interface.
4. Move to the next test case execution.

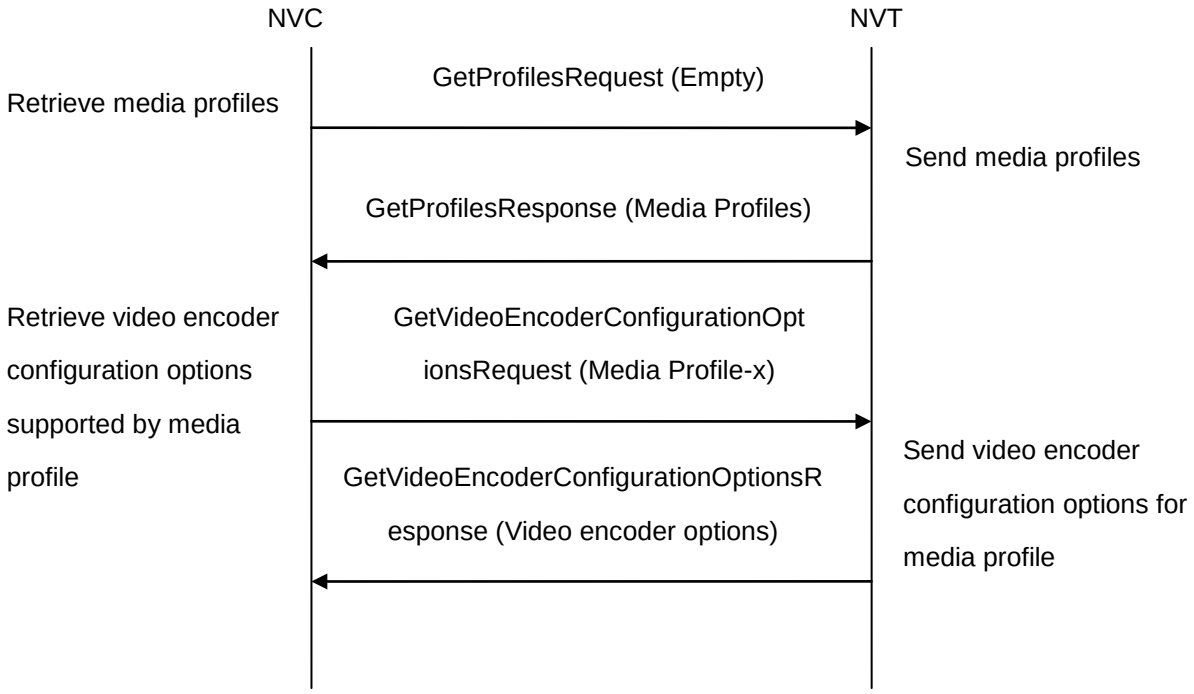


A.14 Media Profile Configuration for Video Streaming

For the execution of real time streaming - video test cases, NVC has to select and configure the media profile based on the required video codec.

NVC follows the following procedure to configure the media profile.

1. Retrieve media profiles by invoking GetProfiles command.
2. Retrieve supported video encoder configuration options for a media profile by invoking GetVideoEncoderConfigurationOptions (**media profile token**) command. Check whether the selected media profile supports the required video codec.
3. Repeat test procedure-2 for all media profiles till a media profile with the required video codec support is found.



A.15 Media Profile Configuration for Audio & Video Streaming

For the execution of real time streaming – Audio & Video test cases, NVC has to select and configure the media profile based on the required audio & video codec.

NVC follows the following procedure to configure the media profile.

1. Retrieve media profiles by invoking GetProfiles command.
2. Retrieve supported video encoder configuration options for a media profile by invoking GetVideoEncoderConfigurationOptions (**media profile token**) command. Check whether the selected media profile supports the required video codec.
3. If the media profile includes audio source and audio encoder configurations

a. Retrieve supported audio encoder configuration options for the media profile by invoking GetAudioEncoderConfigurationOptions (**media profile token**) command. Check whether the selected media profile supports the required audio codec.

4. If the media profile doesn't have audio source and audio encoder configuration

a. Retrieve audio source configurations of NVT by invoking GetAudioSourceConfigurations command.

b. Add audio source configuration to the profile by invoking AddAudioSourceConfigurations command.

c. Retrieve audio encoder configurations of NVT by invoking GetAudioEncoderConfigurations command.

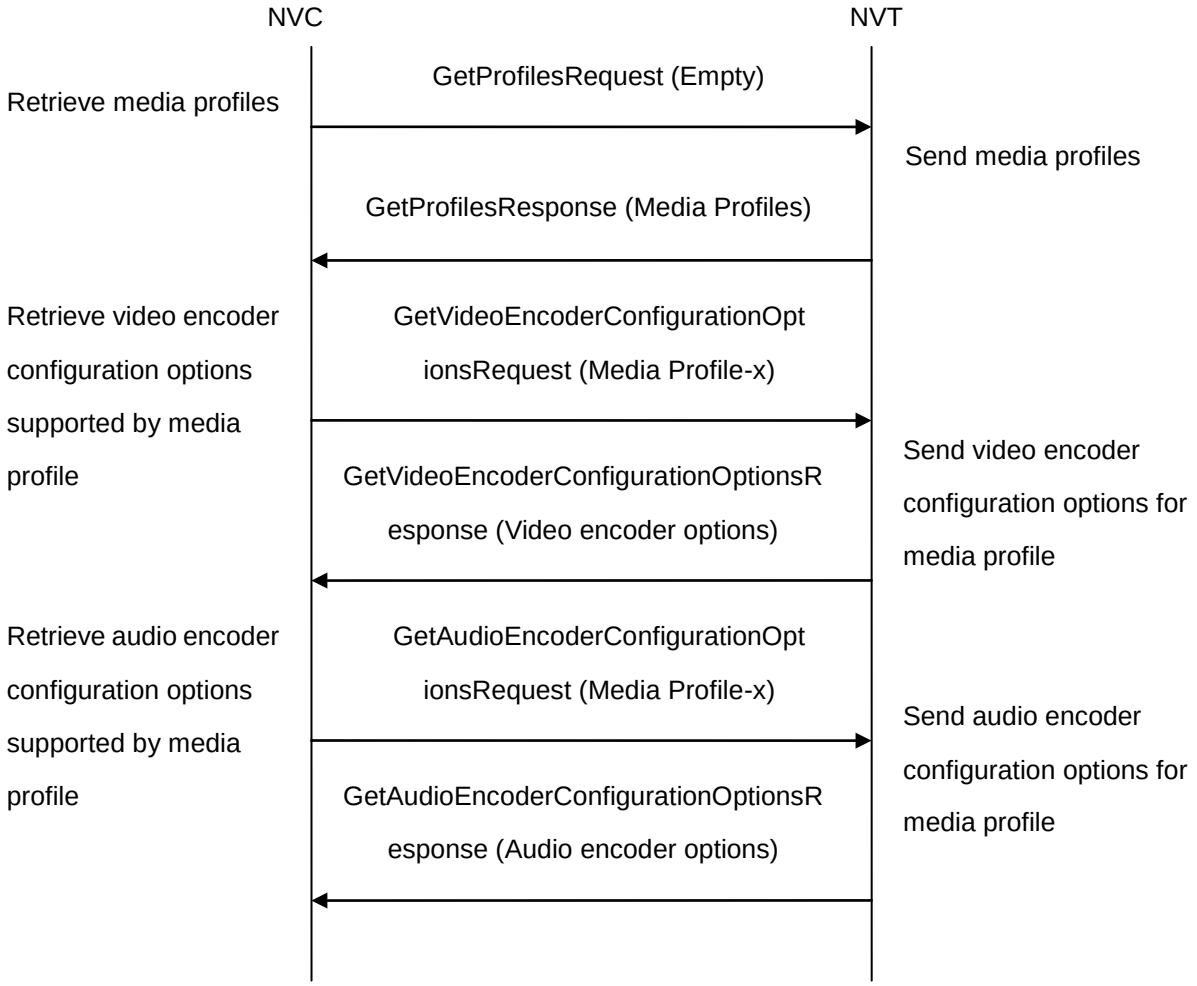
d. Retrieve audio encoder configuration options supported for an audio encoder configuration by invoking GetAudioEncoderConfigurationOptions (**audio encoder config token**) command. Check whether the selected audio encoder configuration supports the required audio codec.

e. Repeat test procedure – 4.d for all audio encoder configurations till a audio encoder configuration with the required audio codec is found.

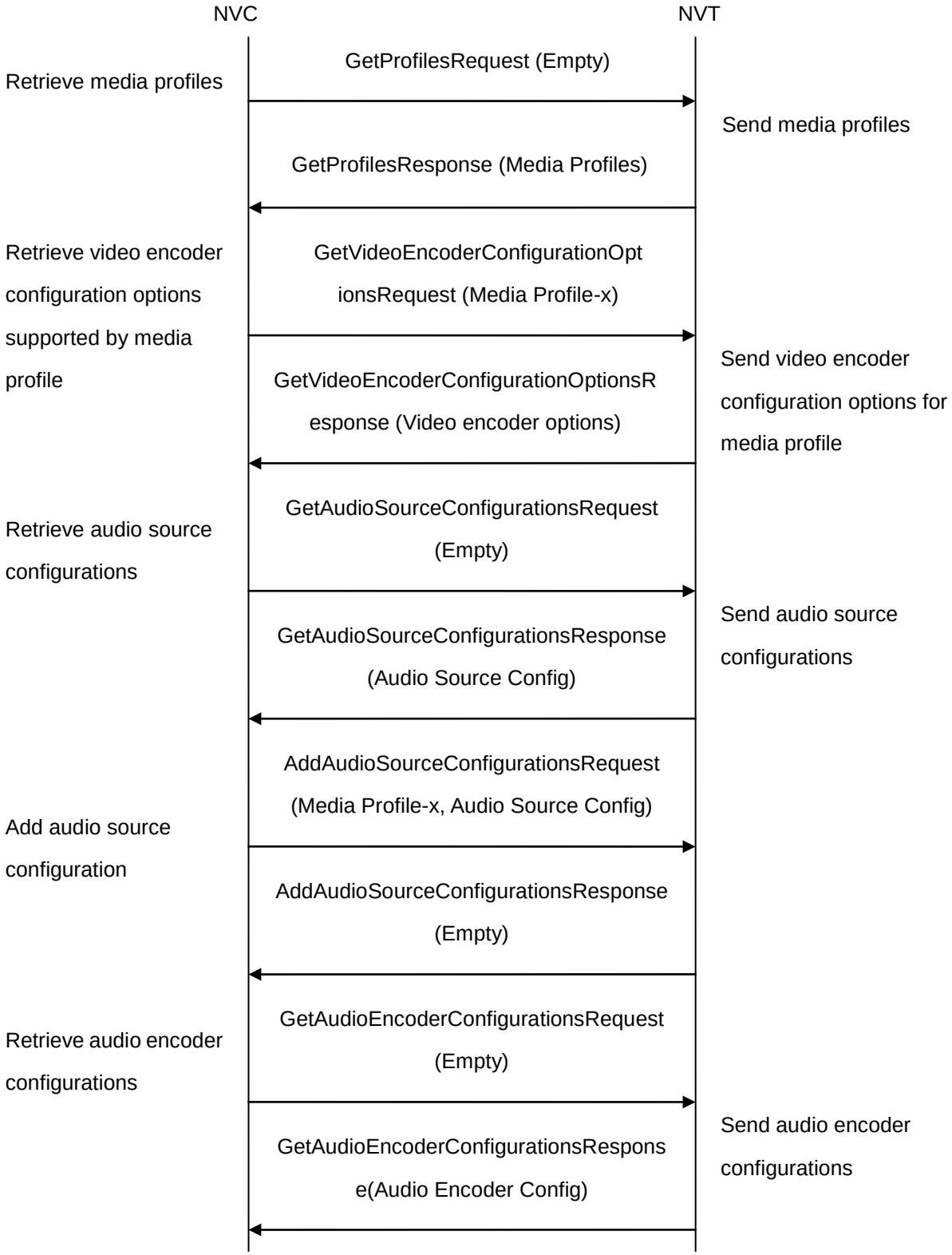
f. Add the corresponding audio encoder configuration to the media profile by invoking AddAudioEncoderConfiguration command.

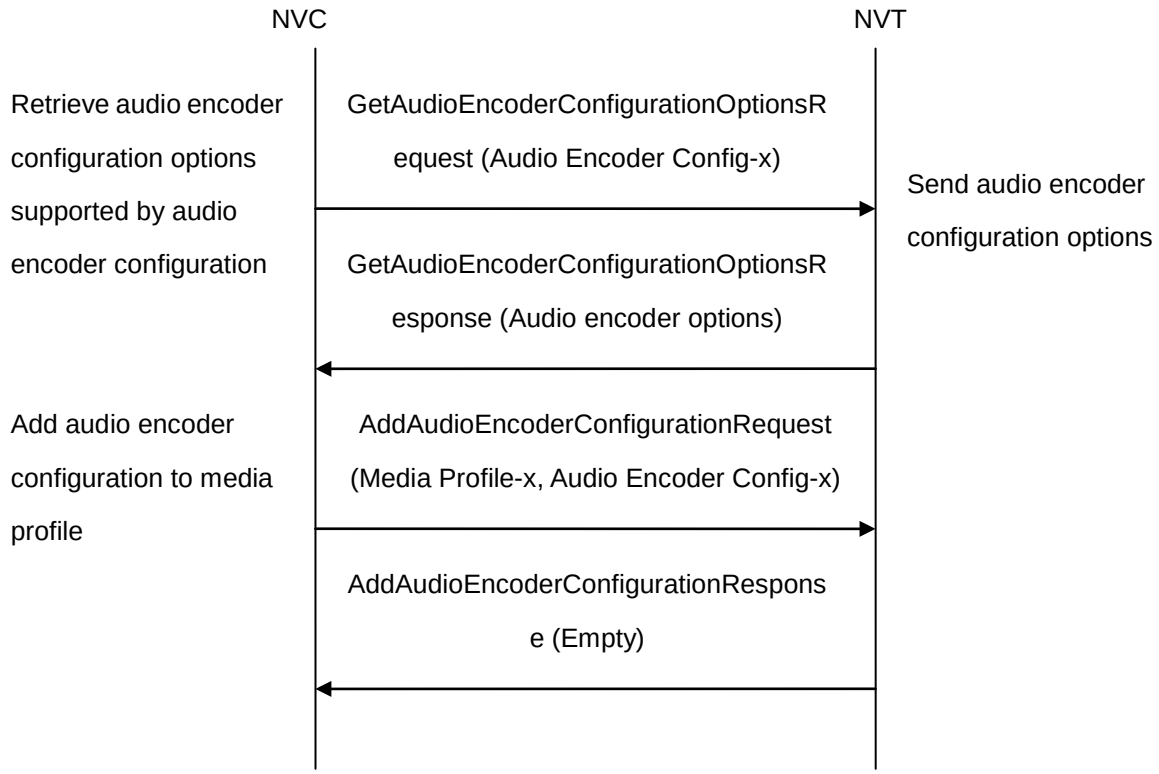
5. Repeat test procedure-2, 3 & 4 for all media profiles till a media profile with the required audio and video codec support is found.

Case 1: Media profile includes audio source and audio encoder configurations:



Case 2: Media profile without audio source and audio encoder configurations:

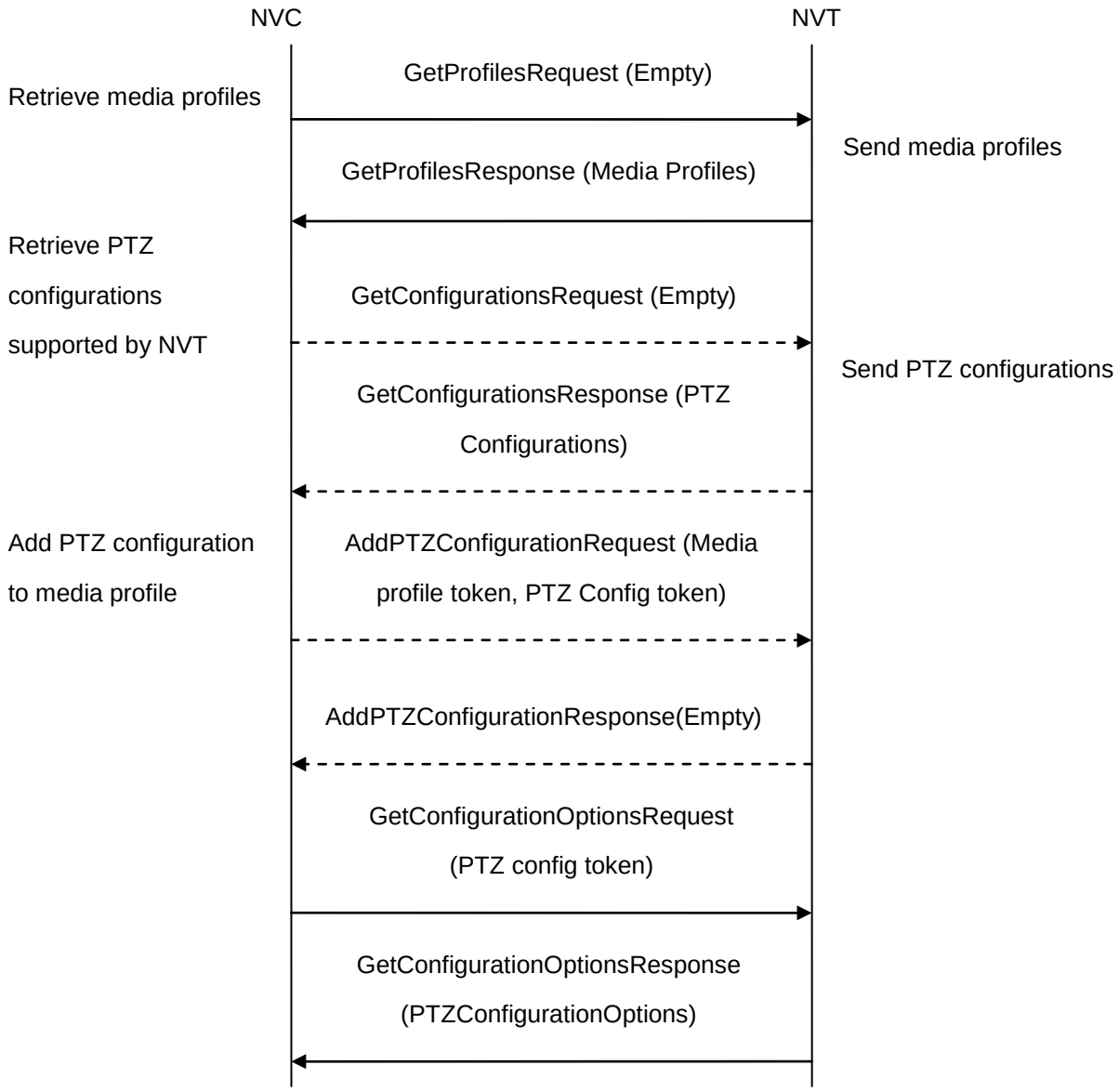



NVC
NVT
Retrieve audio encoder configuration options supported by audio encoder configuration
GetAudioEncoderConfigurationOptionsRequest (Audio Encoder Config-x)
Send audio encoder configuration options
GetAudioEncoderConfigurationOptionsResponse (Audio encoder options)
Add audio encoder configuration to media profile
AddAudioEncoderConfigurationRequest (Media Profile-x, Audio Encoder Config-x)
AddAudioEncoderConfigurationResponse (Empty)

A.16 Media Profile Configuration for PTZ Control

For the execution of PTZ control test cases, NVC has to select and configure the media profile as follows.

1. Retrieve media profiles by invoking GetProfiles command. Check whether a media profile contains PTZ configuration or not.
2. If no media profile contains PTZ configuration, retrieve PTZ configurations by invoking GetConfigurations command.
3. Add PTZ configuration to media profile by invoking AddPTZConfiguration command.
4. Get PTZ configuration options for the added PTZ Configuration by invoking GetConfiguraitonOptions



A.17 MetadataConfiguration for receiving / not receiving events metadata

When receiving metadata a client might be interested in receiving all, none or some of the events produced by NVT. The basic rules for configuring a MetadataConfiguration to achieve this are:

- To get all events: Include the Events element but do not include a filter.
- To get no events: Do not include the Events element.
- To get only some events: Include the Events element and include a filter in the element.

This means that if an Events element is **not** included in a MetadataConfiguration, then **no** Events metadata shall be included in the metadata stream for that configuration. Similarly, if an "<Events>" tag **without any sub-tags** is included in a MetadataConfiguration it means that **all** available Events metadata shall be included in the metadata stream for that configuration.

Example:

The following SetMetadataConfigurationRequest can be used when the client is interested in a metadata stream that includes all Events, but nothing else. The PTZStatus element is not included in the configuration, so no PTZ metadata will be included in the metadata stream. The Events element is included, but without any filter, so all Events will be included in the metadata stream.

```
<SetMetadataConfiguration>
    <Configuration token="Test">
        <Name>TestName</Name>
        <UseCount>1</UseCount>
        <Events/>
        <Analytics>false</Analytics>
    </Configuration>
    <ForcePersistence>true</ForcePersistence>
</SetMetadataConfiguration>
```

A.18 Subscribe and CreatePullpointSubscription for receiving all Events

When subscribing for events a client might be interested in receiving all or some of the Events produced by the NVT.

If a client is interested in receiving some events it includes a filter tag in the CreatePullPointSubscription or Subscribe requests describing the events which the client is interested in (see examples in the Core Spec).

If a client is interested in receiving all events from a device it does not include the Filter sub tag in the Subscribe or CreatePullPointSubscription request.

Example:

The following Subscribe and CreatePullPointSubscription requests can be used if a client is interested in receiving all events.

1) Subscribe Request:

```
<m:Subscribe xmlns:m="http://docs.oasis-open.org/wsn/b-2"
xmlns:m0="http://www.w3.org/2005/08/addressing">
    <m:ConsumerReference>
        <m0:Address>
            http://192.168.0.1/events
        </m0:Address>
        </m:ConsumerReference>
</m:Subscribe>
```

2) CreatePullpointSubscriptionRequest

```
<m:CreatePullPointSubscription xmlns:m="http://www.onvif.org/ver10/events/wsdl"/>
```

A.19 Valid expression indicating empty IP Address

If a certain IP Address is not set (e.g. an NTP or a DNS Address) the device can use one of the following three possibilities:

1) Use 0.0.0.0 as empty IP Address

```
<IPAddress>0.0.0.0</IPAddress>
```

2) Use an empty IP Address tag

```
<IPAddress/>
```

3) Omit the IP Address tag (if it is an optional element)

Example:

1) Use 0.0.0.0

```
<m:GetDNSResponse xmlns:m="http://www.onvif.org/ver10/device/wsdl"
xmlns:m0="http://www.onvif.org/ver10/schema">
    <m:DNSInformation>
        <m0:FromDHCP>false</m0:FromDHCP>
        <m0:DNSManual>
```

```
            <m0:Type>IPv4</m0:Type>
            <m0:IPv4Address>0.0.0.0</m0:IPv4Address>
        </m0:DNSManual>
    </m:DNSInformation>
</m:GetDNSResponse>
```

2) Use an empty tag

```
<m:GetDNSResponse xmlns:m="http://www.onvif.org/ver10/device/wsdl"
xmlns:m0="http://www.onvif.org/ver10/schema">
    <m:DNSInformation>
        <m0:FromDHCP>false</m0:FromDHCP>
        <m0:DNSManual>
            <m0:Type>IPv4</m0:Type>
            <m0:IPv4Address/>
        </m0:DNSManual>
    </m:DNSInformation>
</m:GetDNSResponse>
```

3) Omit tag

```
<m:GetDNSResponse xmlns:m="http://www.onvif.org/ver10/device/wsdl"
xmlns:m0="http://www.onvif.org/ver10/schema">
    <m:DNSInformation>
        <m0:FromDHCP>false</m0:FromDHCP>
        <m0:DNSManual>
            <m0:Type>IPv4</m0:Type>
        </m0:DNSManual>
    </m:DNSInformation>
</m:GetDNSResponse>
```