ONVIF®

Core Client Test Specification

Version 18.12

December 2018

REVISION HISTORY

Vers.	Date	Description
18.12	Dec 11, 2018	The following was done according to #299: Security Test Cases (removed) Username Token Test Cases (added) HTTP Digest Test Cases (added) Annex A.1 Required Number of Devices Summary (updated)
18.12	Oct 3, 2018	The following was changed according to #257: Auxiliary Commands Test Cases (updated) Annex A.1 Required Number of Devices Summary (updated)
18.12	Aug 13, 2018	The following features added according to #278: Network Video Transmitter Discovery Type Filter Device Discovery Type Filter
18.06	Jun 21, 2018	Reformatting document using new template
18.06	Jun 04, 2018	EVENTHANDLING-3 METADATA STREAMING test case updated according to #241
18.06	Apr 05, 2018	'Required Number of Devices Summary' Annex added according to #241
18.06	Feb 13, 2018	The following were updated in the scope of #241: Feature Level Requirement (updated with new rules) Each Feature Level Requirement (updated with Check Condition based on Device Features and Required Number of Devices)
17.12	Aug 15, 2017	Requirement level of Profile T of the following feature was changed from Mandatory to Cconditional according to #220: Auxiliary Commands
17.12	Aug 14, 2017	Profile T Normative Reference and Profile T Normative Reference were added for the following features according to #221: Discovery, Network Configuration, System, User Handling, EVENTHANDLING-3 METADATA STREAMING. Advanced Event Handling feature was divided into Set Synchronization Point feature and Unsubscribe feature according to #220. Profile T Normative Reference of Keep Alive for Pull Point Event Handling were cnaged to Optional according to #220.
17.06	Jun 8, 2017	Profile T Normative Reference and Profile T Requirement Level were added according to #201 in the following secrion: NTP Test Cases

17.06	May 20, 2017	EVENTHANDLING-3 METADATA STREAMING profiles references were updated according #115. Normative references were updated according #187.
17.06	May 16, 2017	Requirement level for Profile S, Profile G, and Profile Q was added into Event Handling feature.
17.06	Mar 31, 2017	Relay Outputs Test Cases were changed according to #188 Get Services Test Cases were added according to #70 EVENTHANDLING-3 METADATA STREAMING test was updated according to #168 Profile T Normative Reference was added for the Capabilities Test Cases
17.06	Mar 24, 2017	Advanced Event Handling Test Cases added. Auxiliary Commands Test Cases added. Scope section updated.
17.06	Mar 20, 2017	HTTP Digest Authentication for RTSP Test Cases added
17.06	Mar 17, 2017	Profile T Normative Reference were added for the following features: Security, Event Handling, Keep Alive for Pull Point Event Handling
17.06	Mar 02, 2017	The follwing test cases were updated according to #84: USERHANDLING-1 CREATE USERS USERHANDLING-3 SET USER USERHANDLING-4 DELETE USERS
16.12	Okt 18, 2016	Features requirement level was added for all features.
16.07	Jun 14, 2016	EVENTHANDLING-3 METADATA STREAMING test case has been updated. Test steps sequence was changed.
16.07	May 11, 2016	Profile Q requirement level was updated for the following test cases: ZEROCONFIGURATION-1, ZEROCONFIGURATION-2 Hostname Configuration Test Cases were added. DNS Configuration Test Cases were added. Network Protocols Configuration Test Cases were added.
16.07	Apr 19, 2016	• Test cases about specific event were removed: MONITORINGNOTIFICATIONS-1, MONITORINGNOTIFICATIONS-2, MONITORINGNOTIFICATIONS-3, MONITORINGNOTIFICATIONS-4, DEVICEMANAGEMENTNOTIFICATIONS-1, DEVICEMANAGEMENTNOTIFICATIONS-2, DEVICEMANAGEMENTNOTIFICATIONS-3, DEVICEMANAGEMENTNOTIFICATIONS-4, DEVICEMANAGEMENTNOTIFICATIONS-5. • Monitoring Notifications scenario updated • Device Management Notifications scenario updated

16.07	Apr 18, 2016	System Date and Time Configuration test cases were updated: Normative References for Profile S, Profile A, Profile C, and Profile G were updated. Step description in Test Procedure was updated for the EVENTHANDLING-3 test case. Old description: Device response has code RTSP 200 OK if it is detected New description: If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK
16.07	Mar 18, 2016	Checking of TEARDOWN response was changed in Test Procedure and PASS criteria for the EVENTHANDLING-3 test case. Old description of checking of TEARDOWN response in Test Procedure: Device responds with code RTSP 200 OK. New description of checking of TEARDOWN response in Test Procedure: Device response has code RTSP 200 OK if it is detected. Old description of checking of TEARDOWN response in PASS criteria: Device response on the RTSP TEARDOWN request fulfills the following requirements: New description of checking of TEARDOWN response in PASS criteria: If there is Device response on the RTSP TEARDOWN request then it fulfills the following requirements:
16.07	Mar 16, 2016	Docbook stylesheets were updated.
16.07	Mar 14, 2016	www.onvif.org was removed from Copyright section.
16.07	Feb 26, 2016	The following steps were removed because the requirements are fullfield by XML Schemas validation: • SET NTP SETTINGS: [S2] "<SetNTP>" includes tag: "<FromDHCP>" with "TRUE" OR "FALSE" value AND • SET ZERO CONFIGURATION SETTINGS: [S3] "<SetZeroConfiguration>" includes tag: "<Enabled>" with "TRUE" OR "FALSE" value AND • GET SERVICES: [S2] (Client request does not contain "<IncludeCapability>" tag OR "<GetServices>" includes tag: "<IncludeCapability>" with either "TRUE" OR "FALSE" values) AND
16.07	Jan 28, 2016	HTTP System Backup Test Cases and HTTP System Restore Test Cases were added.

16.07	Jan 27, 2016	Remote User Handling Test Cases were moved into ONVIF Postponed Test Specification since this functionality was removed from Profile Q
16.07	Jan 21, 2016	RFC 2617 was added to normative reference. OASIS Web Services Security UsernameToken Profile 1.0 was added to normative reference. WS-Discovery was added to normative reference. The following namespaces were added to the list: • http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd • http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd • http://schemas.xmlsoap.org/ws/2005/04/discovery • http://schemas.xmlsoap.org/ws/2004/08/addressing The description about structure and hierarchy was replaced for the test cases: SECURITY-1, CAPABILITIES-1, CAPABILITIES-2, EVENTHANDLING-1, EVENTHANDLING-2, DISCOVERY-1, NETWORKCONFIGURATION-1, NETWORKCONFIGURATION-2, NETWORKCONFIGURATION-3, NETWORKCONFIGURATION-4, SYSTEM-1, USERHANDLING-1, USERHANDLING-2, USERHANDLING-3, USERHANDLING-4, RELAYOUTPUTS-1, RELAYOUTPUTS-2, RELAYOUTPUTS-3, RELAYOUTPUTS-4, NTP-1, NTP-2, DYNAMICDNS-1, DYNAMICDNS-2, ZEROCONFIGURATION-1, ZEROCONFIGURATION-2, IPADDRESSFILTERING-1, IPADDRESSFILTERING-2, IPADDRESSFILTERING-3, IPADDRESSFILTERING-4, IPADDRESSFILTERING-5, IPADDRESSFILTERING-6, IPADDRESSFILTERING-7, PERSISTENTNOTIFICATIONSTORAGERETRIEVAL-1 Old description: Client %COMMAND NAME% request message is a well-formed SOAP request (refer to onvif.xsd) AND Client %COMMAND NAME% request message has a proper hierarchy (refer to %SERVICE%.wsdl) AND New description: Client %COMMAND NAME% request messages are valid according to XML Schemas listed in Namespaces AND Client %COMMAND NAME% request in Test Procedure fulfills the following requirements: The following steps was removed because the requirements are fullfield by XML Schemas validation: • EVENTHANDLING-1: [S5] "<PullMessages>" includes tag: "<Timeout>" AND [S6] "<PullMessages>" includes tag: "<MessageLimit>" AND • EVENTHANDLING-2: [S2] "<Subscribe>" includes tag: "<ConsumerReference>" AND

[S3] "<ConsumerReference>" includes tag: "<Address>" AND

- EVENTHANDLING-2:

  [S2] "<Subscribe>" includes tag: "<ConsumerReference>" AND

  [S3] "<ConsumerReference>" includes tag: "<Address>" AND
- NETWORKCONFIGURATION-2:

  [S3] "<SetNetworkInterfaces>" includes tag: "<NetworkInterface>" AND
- USERHANDLING-1:

  [S5] "<User>" includes tag: "<UserLevel>" with non-empty string value AND
- USERHANDLING-3:

  [S4] "<User>" includes tag: "<UserLevel>" with non-empty string value AND
- RELAYOUTPUTS-2:

  [S3] "<SetRelayOutputState>" includes tag: "<LogicalState>" with "Active" OR "Inactive" value AND
- RELAYOUTPUTS-3:

  [S3] "<SetRelayOutputSettings>" includes tag: "<Properties>" AND

  [S5] "<Properties>" includes tag: "<DelayTime>" AND

  [S6] "<Properties>" includes tag: "<IdleState>" with "Closed" OR "Open" value AND
- RELAYOUTPUTS-4:

  [S3] "<SetRelayOutputSettings>" includes tag: "<Properties>" AND

  [S5] "<Properties>" includes tag: "<DelayTime>" AND

  [S6] "<Properties>" includes tag: "<IdleState>" with "Closed" OR "Open" value AND
- DYNAMICDNS-2:

  [S2] "<SetDynamicDNS>" includes tag: "<Type>" with value EITHER "NoUpdate" OR "ClientUpdates" OR "ServerUpdates" AND
- IPADDRESSFILTERING-2:

  [S2] "<SetIPAddressFilter>" includes tag: "<Type>" with "Allow" OR "Deny" value AND
- IPADDRESSFILTERING-3:

  [S2] "<SetIPAddressFilter>" includes tag: "<Type>" with "Allow" OR "Deny" value AND
- IPADDRESSFILTERING-4:

  [S2] "<AddIPAddressFilter>" includes tag: "<Type>" with "Allow" OR "Deny" value AND
- IPADDRESSFILTERING-5:

  [S2] "<AddIPAddressFilter>" includes tag: "<Type>" with "Allow" OR "Deny" value AND
- IPADDRESSFILTERING-6:

		[S2] "<RemoveIPAddressFilter>" includes tag: "<Type>" with "Allow" OR "Deny" value AND • IPADDRESSFILTERING-7: [S2] "<RemoveIPAddressFilter>" includes tag: "<Type>" with "Allow" OR "Deny" value AND • PERSISTENTNOTIFICATIONSTORAGERETRIEVAL-1: [S5] "<Seek>" includes tag: "<UtcTime>" with non-empty value of date and time AND [S9] "<PullMessages>" includes tag: "<Timeout>" AND [S10] "<PullMessages>" includes tag: "<MessageLimit>" AND
16.07	Dec 30, 2015	METADATA STREAMING test case was updated to check of media type in RTSP SETUP requests and to check of corresponding between RTSP session and GetStreamUri. Device Management Notifications was added.
16.07	Dec 24, 2015	Monitoring Notifications was added.
16.07	Dec 23, 2015	System Date and Time Configuration was added. Remote User Handling was added. HTTP Firmware Upgrade was added. Normative references were updated.
16.01	Dec 08, 2015	Keep Alive for Pull Point Event Handling Test Cases feture failed criteria were updated New precondition was added to GETSERVICES-1.
16.01	Dec 03, 2015	General item (Test Owerview) was added Minor updates in formatting, typos and terms. Keep Alive for Pull Point Event Handling Test Cases was updated to remove verification of Action and ReferenceParameters.
16.01	Jen 08, 2016	Advanced Pull Point Event Handling was added. Profile A requirement level was added for old test cases. Get Services with Capabilities was added.
15.06	Jun 10, 2015	No major changes were made, just minor formatting fixes.
15.05	May 20, 2015	No major changes were made, just minor grammatical corrections.
15.03	Mar 20, 2015	Added new Test Cases sections: Discovery, Network Configuration, System, User Handling, Relay Outputs, NTP, Dynamic DNS, Zero Configuration, IP Address Filtering and Persistent Notification Storage Retrieval.
14.12	Dec 11, 2014	Fixed typos and inconsistencies.
14.11	Nov 21, 2014	Fixed typos and inconsistencies. Removed examples of expected Requests and Responses from all Test Cases. Removed unnecessary PASS criteria from all Test Cases.

		EVENTHANDLING-1 and EVENTHANDLING-3 test cases have been updated. "3. Terms and Definitions" section has been updated. Introduced YY.MM method of version numbering
1.4	Sep 04, 2014	The SECURITY-1 USER TOKEN PROFILE test case has been updated. The SECURITY-2 DIGEST AUTHENTICATION test case has been updated. The CAPABILITIES-1 GET SERVICES test case has been updated. The CAPABILITIES-2 GET CAPABILITIES test case has been updated. The EVENTHANDLING-1 PULLPOINT test case has been updated. The EVENTHANDLING-2 BASE NOTIFICATION test case has been updated. The EVENTHANDLING-3 METADATA STREAMING test case has been updated. "Scope", "Security", "Capabilities" and "Event Handling" sections have been updated.
1.3	Jul 31, 2014	The SECURITY-1 USER TOKEN PROFILE test case has been updated. The SECURITY-2 DIGEST AUTHENTICATION test case has been updated. Section "Test Policy" has been removed. "Introduction", "Scope", "Security", "Capabilities", "Event Handling", "Normative references", "Definition" and "Test Setup" sections have been updated. The CAPABILITIES-1 GET SERVICES test case has been added. The CAPABILITIES-2 GET CAPABILITIES test case has been added. The EVENTHANDLING-1 PULLPOINT test case has been added. The EVENTHANDLING-2 BASE NOTIFICATION test case has been added. The EVENTHANDLING-3 METADATA STREAMING test case has been added.
1.2	Jun 27, 2014	Subsections "Capabilities" and "Event Handling" have been added to "Introduction" section. "Definition" section has been updated. "Test Setup" section has been updated. Subsections "Capabilities" and "Event Handling" have been added to "Test Policy" section. Tests "GET SERVICES" and "GET CAPABILITIES" have been added to "Capabilities Test Cases" section.

		Tests "PULLPOINT", "BASE NOTIFICATION" and "METADATA STREAMING" have been added to "Event Handling Test Cases" section. Examples of expected Requests and Responses have been updated for "Security Test Cases" section.
1.1	Jun 16, 2014	Changes were made in the Security Test Cases specification. The new section "Normative references" has been added. "Introduction", "Scope" and "Security" sections have been updated. "Definition" section has been updated.
1.0	Jun 11, 2014	Initial version

Table of Contents

1 Introduction

The goal of the ONVIF Test Specification set is to make it possible to realize fully interoperable IP physical security implementations from different vendors. This specification also acts as an input document to the development of a test tool which will be used to test the ONVIF Client implementation conformance towards ONVIF standard. This Client Test Tool analyzes network communications between ONVIF Devices and Clients being tested and determines whether a specific Client is ONVIF conformant (see ONVIF Conformance Process Specification).

This particular document defines test cases required for testing Core features of a Client application e.g. EventHandling, Security and Capabilities. Also the test cases are to be basic inputs for some Profile specification requirements. It also describes the test framework, test setup, pre-requisites, test policies needed for the execution of the described test cases.

1.1 Scope

This ONVIF Core Client Test Specification defines and regulates the conformance testing procedure for the ONVIF conformant Clients in the scope of Core features. Conformance testing is meant to be black-box network traces analysis and verification. The objective of this specification is to provide the test cases to test individual requirements of ONVIF Clients in the scope of Core features according to ONVIF Profile Specifications.

The principal intended purposes are:

- Provide self-assessment tool for implementations.
- Provide comprehensive test suite coverage for Core features.

This specification **does not** address the following:

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

The following sections cover test cases needed for the verification of relevant features as mentioned in the ONVIF Profile Specifications.

1.2 Username Token

Username Token section defines security mechanism for Username Token Profile.

1.3 HTTP Digest

HTTP Digest section defines security mechanism for HTTP Digest Authentication.

1.4 Capabilities

Capabilities section specifies Client ability to retrieve available services and advanced functionalities which are offered by a Device.

1.5 Get Services with Capabilities

Get Services with Capabilities section specifies Client ability to retrieve capabilities of services with using GetServices operation.

1.6 Event Handling

Event Handling section defines Client ability to initiate and receive notifications (events) from a Device.

The event handling test cases cover the following mandatory interfaces:

- Pull Point Notification Interface
  - This test specification provides test cases to verify the implementation of the PullPoint Interface of a Client.
- Basic Notification Interface
  - This test specification provides test cases to verify the implementation of the Basic Notification Interface of a Client.
- Metadata Streaming Interface
  - This test specification provides test cases to verify the implementation of the Metadata Streaming Interface of a Client.

1.7 Set Synchronization Point

Set Synchronization Point section defines Client ability to synchronize its properties with the properties of the device using SetSynchronizationPoint operation.

1.8 Unsubscribe

Unsubscribe section defines Client ability to terminete subscription using Unsubscribe operation.

1.9 Keep Alive for Pull Point Event Handling

Keep Alive for Pull Point Event Handling section specifies Client ability to use keep alive for Pull Point Event Handling using PullMessages or Renew approach.

1.10 Discovery

Discovery section defines Client ability to locate services on a local network using Web Services Dynamic Discovery (WS-Discovery) protocol. It uses IP multicast address 239.255.255.250 and TCP and UDP port 3702 and SOAP-over-UDP standard for communication between nodes.

1.11 Network Configuration

Network Configuration section defines Client ability to obtain and configure of network settings on Device.

1.12 System

System section defines Client ability to obtain Device information and configure of system settings on Device.

1.13 User Handling

User Handling section defines Client ability to manage users on Device.

1.14 Relay Outputs

Relay Outputs section defines Client ability to list, configure and trigger relay outputs on Device.

1.15 NTP

NTP section defines Client ability to configure synchronization of time using NTP servers on Device.

1.16 Dynamic DNS

Dynamic DNS section defines Client ability to configure dynamic DNS settings on Device.

1.17 Zero Configuration

Zero Configuration section defines Client ability to enable or disable zero configuration on Device.

1.18 IP Address Filtering

IP Address Filtering section defines Client ability to manage IP address filters on Device.

1.19 Persistent Notification Storage Retrieval

Persistent Notification Storage Retrieval section defines Client ability to seek stored events in Device.

1.20 System Date and Time Configuration

System Date and Time Configuration section defines Client ability to configure Device system date and time using GetSystemDateAndTime and SetSystemDateAndTime operations.

1.21 HTTP Firmware Upgrade

HTTP Firmware Upgrade section defines Client ability to upgrade Device firmware over HTTP using StartFirmwareUpgrad operation and HTTP POST.

1.22 HTTP System Backup

HTTP System Backup section defines Client ability to backup system configurations over HTTP using GetSystemUris operation and HTTP GET.

1.23 HTTP System Restore

HTTP System Restore section defines Client ability to restore system configurations over HTTP using StartSystemRestore operation and HTTP POST.

1.24 Monitoring Notifications

Monitoring Notifications section specifies Client ability to receive from Device monitoring notifications.

1.25 Device Management Notifications

Device Management Notifications section specifies Client ability to receive from Device device management notifications.

1.26 Hostname Configuration

Hostname Configuration section defines Client ability to obtain and configure of hostname settings on Device.

1.27 DNS Configuration

DNS Configuration section defines Client ability to obtain and configure of DNS settings on Device.

1.28 Network Protocols Configuration

Network Protocols Configuration section defines Client ability to obtain and configure of network protocols settings on Device.

1.29 HTTP Digest Authentication for RTSP

HTTP Digest Authentication for RTSP section defines security mechanism for Digest Authentication for RTSP.

1.30 Auxiliary Commands

Auxiliary Commands section defines Client ability to manage auxiliary commands supported by the Device.

2 Normative references

- ONVIF Conformance Process Specification:

  https://www.onvif.org/profiles/conformance/

- ONVIF Profile Policy:

  https://www.onvif.org/profiles/

- ONVIF Core Specifications:

  https://www.onvif.org/profiles/specifications/

- ONVIF Streaming Specification:

  https://www.onvif.org/profiles/specifications/

- ONVIF Profile S Specification:

  https://www.onvif.org/profiles/profile-s/

- ONVIF Profile G Specification:

  https://www.onvif.org/profiles/profile-g/

- ONVIF Profile C Specification:

  https://www.onvif.org/profiles/profile-c/

- ONVIF Profile Q Specification:

  https://www.onvif.org/profiles/profile-q/

- ONVIF Profile A Specification:

  https://www.onvif.org/profiles/profile-a/

- ONVIF Core Client Test Specification:

  https://www.onvif.org/profiles/conformance/client-test/

- ISO/IEC Directives, Part 2, Annex H:

  http://www.iso.org/directives

- ISO 16484-5:2014-09 Annex P:

  https://www.iso.org/obp/ui/#!iso:std:63753:en

- WS-BaseNotification:

  http://docs.oasis-open.org/wsn/wsn-ws_base_notification-1.3-spec-os.pdf

- W3C SOAP 1.2, Part 1, Messaging Framework:

  http://www.w3.org/TR/soap12-part1/

- W3C XML Schema Part 1: Structures Second Edition:

  http://www.w3.org/TR/xmlschema-1/

- W3C XML Schema Part 2: Datatypes Second Edition:

  "http://www.w3.org/TR/xmlschema-2/ [http://www.w3.org/TR/xmlschema-2/]

- W3C Web Services Addressing 1.0 – Core:

  http://www.w3.org/TR/ws-addr-core/

- OASIS Web Services Security UsernameToken Profile 1.0:

  http://docs.oasis-open.org/wsn/wsn-ws_topics-1.3-spec-os.pdf

- IETF RFC 2617, HTTP Authentication:

  http://www.ietf.org/rfc/rfc2617.txt

- XMLSOAP, Web Services Dynamic Discovery (WS-Discovery), J. Beatty et al., April 2005.

  http://specs.xmlsoap.org/ws/2005/04/discovery/ws-discovery.pdf

3 Terms and Definitions

3.1 Conventions

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

3.2 Definitions

This section describes terms and definitions used in this document.

Address	An address refers to a URI.
Profile	See ONVIF Profile Policy.
ONVIF Device	Computer appliance or software program that exposes one or multiple ONVIF Web Services.
ONVIF Client	Computer appliance or software program that uses ONVIF Webservices.
Capability	List of services and features supported by an ONVIF Device.
Metadata	All streaming data except video and audio, including video analytics results, PTZ position data and other metadata (such as textual data from POS applications).
Conversation	A conversation is all exchanges between two MAC addresses that contains SOAP request and response.
Network	A network is an interconnected group of devices communicating using the Internet protocol.
Network Trace Capture file	Data file created by a network protocol analyzer software (such as Wireshark). Contains network packets data recorded during a live network communications.
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.
Client Test Tool	ONVIF Client Test Tool that tests ONVIF Client implementation towards the ONVIF Test Specification set.
NO SOAP ERROR	Indication of absence of a SOAP Fault element (which is used to indicate error messages). If a Fault element is present, it shall appear as a child element of the Body element. A Fault element can only appear once in a SOAP message.
Valid Device Response	Device has responded to specific request with code HTTP or RTSP 200 OK and SOAP fault message has not appeared.
WS-Discovery	Web service specification defines a multicast discovery protocol to locate services. By default, Client sends probes

	to a multicast group, and target services that match return a response directly to the requester.
Zero Configuration	Technology that allows automatically create a computer network over TCP/IP protocol suite between interconnected network units.

3.3 Abbreviations

This section describes abbreviations used in this document.

HTTP	Hyper Text Transport Protocol.
HTTPS	Hyper Text Transport Protocol over Secure Socket Layer.
IP	Internet Protocol.
IPv4	Internet Protocol version 4.
RTCP	RTP Control Protocol.
RTSP	Real Time Streaming Protocol.
SDP	Session Description Protocol.
TCP	Transport Control Protocol.
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.

3.4 Namespaces

Prefix and namespaces used in this test specification are listed in Table 1. These prefixes are not part of the standard and an implementation can use any prefix.

Table 3.1. Defined namespaces in this specification

Prefix	Namespace URI	Description
soapenv	http://www.w3.org/2003/05/soap-envelope	Envelope namespace as defined by SOAP 1.2 [SOAP 1.2, Part 1].
ter	http://www.onvif.org/ver10/error	The namespace for ONVIF defined faults.
xs	http://www.w3.org/2001/XMLSchema	Instance namespace as defined by XS [XML-Schema, Part1] and [XMLSchema,Part 2].
xsi	http://www.w3.org/2001/XMLSchema-instance	XML schema instance namespace.

Prefix	Namespace URI	Description
tns1	http://www.onvif.org/ver10/topics	The namespace for the ONVIF topic namespace.
tt	http://www.onvif.org/ver10/schema	ONVIF XML schema descriptions.
tds	http://www.onvif.org/ver10/device/wsdl	The namespace for the WSDL device service.
tev	http://www.onvif.org/ver10/events/wsdl	The namespace for the WSDL event service.
tas	http://www.onvif.org/ver10/advancedsecurity/wsdl	The namespace for the WSDL advanced security service.
wsnt	http://docs.oasis-open.org/wsn/b-2	Schema namespace of the [WS-BaseNotification] specification.
wsa	http://www.w3.org/2005/08/addressing	Device addressing namespace as defined by [WS-Addressing].
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd	Web Services Security UsernameToken Profile namespace as defined by [OASIS Web Services Security UsernameToken Profile 1.0].
wsu	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-utility-1.0.xsd	Web Services Security utility namespace as defined by [OASIS Web Services Security UsernameToken Profile 1.0].
d	http://schemas.xmlsoap.org/ws/2005/04/discovery	Device discovery namespace as defined by [WS-Discovery].
wsadis	http://schemas.xmlsoap.org/ws/2004/08/addressing	Device addressing namespace referred in WS-Discovery [WS-Discovery].
dn	http://www.onvif.org/ver10/network/wsdl	The namespace used for the remote device discovery service.

4 Test Overview

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

Conformance to ONVIF Core Client Test Specification is a prerequisite which is required for testing Client to conformance with Profile S, G and C.

4.1 General

Test Cases are grouped depending on features. Each Test Cases group provides description of feature requirement level for Profiles, expected scenario under test and related test cases:

- Feature Level Requirement
- Expected Scenarios Under Test
- List of Test Cases

4.1.1 Feature Level Requirement

Feature Level Requirement item contains a feature ID, check condition based on Device features, required number of Devices and feature requirement level for the Profiles, which will be used for Profiles conformance.

To claim this Feature as supported Client shall pass Expected Scenario Under Test:

- for each Device, which supports Device Features defined in Check Condition Based on Device Features
- for at least with number of Devices specified in Required Number of Devices

If Feature Level Requirement is defined as Mandatory for some Profile, Client shall support this Feature to claim this Profile Conformance.

4.1.2 Expected Scenarios Under Test

Expected Scenarios Under Test item contains expected scenario under test, conditions when the feature will be defined as supported and as not supported.

4.1.3 Test Cases

Test Case items contain list of test cases which are related to feature. Test cases provide exact procedure of testing feature support conditions.

Each Test Case contains the following parts:

- Test Label - Unique label for each test
- Test Case ID - Unique ID for each test
- Profile Normative References - Requirement level for the feature under test is defined in Profile Specification. This reference is informative and will not be used in conformance procedure.
- Feature Under Test - Feature which is under current test. Typically a particular command or an event.
- Test Purpose - The purpose of current test case.
- Pre-Requisite - The pre-requisite defines when the test should be performed. In case if pre-reqiusite does not match, the test result will be NOT DETECTED.
- Test Procedure - scenario expected to be reflected in network trace file.
- Test Result - Passed and failed criteria of the test case. Depending on these criteria test result will be defined as PASSED or FAILED.
- Validated Feature List - list of features ID related to this test case.

4.2 Test Setup

Collect Network Traces files required by the test cases.

Collect Feature List XML files for Devices detected in the Network Trace files.

Client shall support all mandatory and conditional features listed in the Device Feature List XML file supplied for the Profiles supported by the Client.

For compatibility with the Core Features, the ONVIF Client shall follow the requirements of the conformance process. For details please see the latest ONVIF Conformance Process Specification.

4.3 Prerequisites

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

The Device shall be configured with an IPv4 address.

The Device shall be able to be discovered by the Client.

5 Username Token Test Cases

5.1 Feature Level Requirement:

Validated Feature: WS-Username Token Authentication

Check Condition based on Device Features: WS-Username Token

Required Number of Devices: 1 (Note: Username Token feature shall be passed with at least one Device and can by not detected with other devices with supporting of WS-Username Token)

Profile S Requirement: Mandatory

Profile A Requirement: None

Profile C Requirement: None

Profile G Requirement: None

Profile Q Requirement: None

Profile T Requirement: None

5.2 Expected Scenarios Under Test:

1. Client invokes a specific command which requires authentication with WS-Username Token authentication header.
2. Device sends a valid response to this request.
3. Client is considered as supporting WS-Username Token if the following conditions are met:
   - Device returns a valid response to specific request with UsernameToken authentication header.
4. Client is considered as NOT supporting WS-Username Token if the following is TRUE:
   - All UsernameToken attempts detected are failed.

5.3 USERNAME TOKEN

Test Label: Security - User token profile

Test Case ID: USERTOKENPROFILE-1

Profile S Normative Reference: Mandatory

Profile G Normative Reference: None

Profile C Normative Reference: None

Profile Q Normative Reference: None

Profile A Normative Reference: None

Profile T Normative Reference: None

Feature Under Test: Security

Test Purpose: To verify that the Client supports the User Token Profile for Message level security.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with UsernameToken Authentication present.

Test Procedure (expected to be reflected in network trace file):

1. Client sends a request (e.g. GetUsers) to the Device with correctly formatted UsernameToken.
2. Verify that the Device accepts the correct request.

Test Result:

PASS -

- Client request messages are valid according to XML Schemas listed in Namespaces AND
- Client request that contains UsernameToken authentication in SOAP header fulfills the following requirements:
  - [S1] Client request contains "<Security>" tag after the "<Header>" tag AND
  - [S2] "<Security>" includes tag: "<UsernameToken>" AND
  - [S3] "<UsernameToken>" includes tag: "<Username>" AND
  - [S4] "<UsernameToken>" includes tag: "<Password>" AND
  - [S5] "<UsernameToken>" includes tag: "<Nonce>" AND
  - [S6] "<UsernameToken>" includes tag: "<Created>" AND

- [S7] Device response contains "HTTP/* 200 OK" AND
- [S8] Device response does NOT contain "<Fault>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: UsernameToken.UsernameTokenAuthentication

6 HTTP Digest Test Cases

6.1 Feature Level Requirement:

Validated Feature: HTTP Digest authentication

Check Condition based on Device Features: Digest

Required Number of Devices: 3

Profile A Requirement: Mandatory

Profile C Requirement: Mandatory

Profile G Requirement: Mandatory

Profile Q Requirement: Mandatory

Profile S Requirement: Mandatory

Profile T Requirement: Mandatory

6.2 Expected Scenarios Under Test:

1. Client invokes a specific command which is under testing without any user credentials (no UsernameToken, no HTTP Digest authentication header).
2. Device returns HTTP 401 Unauthorized error along with WWW-Authentication: Digest header.
3. Client re-sends request with HTTP Digest Authentication header corresponding to header provided in device response.
4. Device sends a valid response to this request.
5. Client is considered as supporting HTTP Digest if the following conditions are met:
   - Device returns a valid response to specific request with HTTP Digest authentication header.
6. Client is considered as NOT supporting HTTP Digest if the following is TRUE:
   - All HTTP Digest attempts detected are failed.

6.3 HTTP DIGEST

Test Label: Security - HTTP Digest Authentication.

Test Case ID: HTTPDIGEST-1

Profile S Normative Reference: Mandatory

Profile G Normative Reference: Mandatory

Profile C Normative Reference: Mandatory

Profile Q Normative Reference: Mandatory

Profile A Normative Reference: Mandatory

Profile T Normative Reference: Mandatory

Feature Under Test: Security

Test Purpose: To verify that the Client supports the HTTP Digest Authentication for HTTP level security.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with HTTP Digest Authentication present.

Test Procedure (expected to be reflected in network trace file):

1. Client sends a request that requires authentication (e.g. GetUsers) to the Device without any authentication.
2. Device rejects the request with HTTP error code 401 AND an HTTP Digest challenge.
3. Client sends a valid request with HTTP Digest Authentication.
4. Device accepts the correct request with response code HTTP 200 OK.

Test Result:

PASS -

- [S1] Client request contains (HTTP GET method OR HTTP POST method) without any authentication AND
- Client HTTP GET request has a proper hierarchy (refer to [RFC 1945]) AND
  - [S2] Device response contains "HTTP/* 401 Unauthorized" AND
  - [S3] Device response contains "realm=*" element AND
  - [S4] Device response contains "nonce=*" element AND

- [S5] Client request contains (HTTP GET method OR HTTP POST method) with "Authorization: Digest username=*" element AND

- Client HTTP GET request with HTTP Authentication has a proper hierarchy (refer to [RFC 1945]) AND
  - [S6] Client request contains "realm=*" element with value from Device response AND
  - [S7] Client request contains "nonce=*" element with value from Device response AND
  - [S8] Client request contains "uri=*" element AND
  - [S9] Device response contains "HTTP/* 200 OK".

FAIL -

- The Client failed PASS criteria.

Validated Feature List: HTTPDigest.HTTPDigestAuthentication

7 Capabilities Test Cases

7.1 Feature Level Requirement:

Validated Feature: Capabilities

Check Condition based on Device Features: None

Required Number of Devices: 3

Profile A Requirement: Mandatory

Profile C Requirement: Mandatory

Profile G Requirement: Mandatory

Profile Q Requirement: Mandatory

Profile S Requirement: Mandatory

Profile T Requirement: Mandatory

7.2 Expected Scenarios Under Test:

1. Client invokes a specific Capabilities command which is under testing.
2. Client is considered as supporting Capabilities if the following conditions are met:
   - Device returns a valid response to GetServices request OR
   - Device returns a valid response to GetCapabilities request.
3. Client is considered as NOT supporting Capabilities if the following is TRUE:
   - No Valid Device Response to GetServices request AND
   - No Valid Device Response to GetCapabilities request.

7.3 GET SERVICES

Test Label: Capabilities - Determine the available Services

Test Case ID: CAPABILITIES-1

Profile S Normative Reference: Mandatory

Profile G Normative Reference: Mandatory

Profile C Normative Reference: Mandatory

Profile Q Normative Reference: Mandatory

Profile A Normative Reference: Mandatory

Profile T Normative Reference: Mandatory

Feature Under Test: Capabilities

Test Purpose: To verify that Device Capabilities is received using GetServices request.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetServices command present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetServices request message to retrieve all services of the Device.
2. Verify that GetServicesResponse message from the Device contains code HTTP 200 OK without SOAP Fault.

Test Result:

PASS -

- Client **GetServices** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetServices** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<GetServices>" tag after the "<Body>" tag AND
  - [S2] Device response contains "HTTP/* 200 OK" AND
  - [S3] Device response contains "<GetServicesResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: Capabilities.GetServicesRequest

7.4 GET CAPABILITIES

Test Label: Capabilities - Get Device Capabilities

Test Case ID: CAPABILITIES-2

Profile S Normative Reference: Mandatory

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Profile T Normative Reference: None

Feature Under Test: Capabilities

Test Purpose: To verify that Device Capabilities is received using GetCapabilities request.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetCapabilities command present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetCapabilities request message to retrieve Device Capabilities of the Device.
2. Verify that GetCapabilitiesResponse response message from the Device contains code HTTP 200 OK without SOAP Fault.

Test Result:

PASS -

- Client **GetCapabilities** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetCapabilities** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<GetCapabilities>" tag after the "<Body>" tag AND
  - [S2] Device response contains "HTTP/* 200 OK" AND
  - [S3] Device response contains "<GetCapabilitiesResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: Capabilities.GetCapabilities

8 Get Services Test Cases

8.1 Feature Level Requirement:

Validated Feature: GetServices

Check Condition based on Device Features: GetServices is supported by Device.

Required Number of Devices: 3

Profile A Requirement: Mandatory

Profile C Requirement: Mandatory

Profile G Requirement: Mandatory

Profile Q Requirement: Mandatory

Profile T Requirement: Mandatory

8.2 Expected Scenarios Under Test:

1. Client connects to Device to retrieve a services using **GetServices** commad.
2. Client is considered as supporting Get Services if the following conditions are met:
   - Client supports Capabilities.GetServicesRequest feature.
3. Client is considered as NOT supporting Get Services if ANY of the following is TRUE:
   - Client does not support Capabilities.GetServicesRequest feature.

9 Get Services with Capabilities Test Cases

9.1 Feature Level Requirement:

Validated Feature: GetServices

Check Condition based on Device Features: GetServices is supported by Device.

Required Number of Devices: 1

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile G Requirement: Optional

Profile Q Requirement: Optional

9.2 Expected Scenarios Under Test:

1. Client connects to Device to retrieve a service capabilities.
2. Client is considered as supporting Get Services with Capabilities if the following conditions are met:
   - Client is able to retrieve a services capabilities using **GetServices** operation.
3. Client is considered as NOT supporting Get Services with Capabilities if ANY of the following is TRUE:
   - No valid responses for **GetServices** request.

9.3 GET SERVICES

Test Label: Get Services with Capabilities - Get Services

Test Case ID: GETSERVICES-1

Profile A Normative Reference: Optional

Profile C Normative Reference: Optional

Profile G Normative Reference: Optional

Profile Q Normative Reference: Optional

Feature Under Test: Get Services

Test Purpose: To verify that services capabilities provided by Device is received by Client using the **GetServices** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetServices** operation with **tds:IncludeCapability** element equal to true present.
- The Device supportes GetServices command.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetServices** request message with **tds:IncludeCapability** element equal to true to retrieve redential service capabilities from the Device.
2. Device responds with code HTTP 200 OK and **GetServicesResponse** message.

Test Result:

PASS -

- Client **GetServices** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetServices** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:GetServices** AND
  - [S2] It contains **tds:IncludeCapability** element equal to true AND
- Device response on the **GetServices** request fulfills the following requirements:
  - [S3] It has HTTP 200 response code AND
  - [S4] **soapenv:Body** element has child element **tds:GetServicesResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: GetServicesWithCapabilities.GetServicesWithCapabilitiesRequest

10 Event Handling Test Cases

10.1 Feature Level Requirement:

Validated Feature: EventHandling

Check Condition based on Device Features: None

Required Number of Devices: 3

Profile S Requirement: Conditional

Profile G Requirement: Conditional

Profile Q Requirement: Conditional

Profile A Requirement: Mandatory

Profile C Requirement: Mandatory

Profile T Requirement: Mandatory

Profile T Requirement: Mandatory

10.2 Expected Scenarios Under Test:

1. Client connects to Device to initiate Event Handling.
2. Client is considered as supporting Event Handling if the following conditions are met:
   - Client is able to handle the Pull Point Event mechanism OR
   - Client is able to handle the Base Notification Event mechanism OR
   - Client is able to handle the Metadata Streaming.
3. Client is considered as NOT supporting Event Handling if the following is TRUE:
   - All Pull Point attempts detected have failed AND
   - All Base Notification attempts detected have failed AND
   - All Metadata Streaming attempts detected have failed.

10.3 PULLPOINT

Test Label: Event Handling - Pull Point

Test Case ID: EVENTHANDLING-1

Profile S Normative Reference: Conditional

Profile G Normative Reference: Conditional

Profile C Normative Reference: Governed by business rule #3

Profile Q Normative Reference: Conditional

Profile A Normative Reference: Mandatory

Profile T Normative Reference: Mandatory

Profile T Normative Reference: Mandatory

Feature Under Test: Event Handling

Test Purpose: To verify that the Client is able to retrieve events using Pull Point.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with Pull Point event type.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes CreatePullPointSubscription message.
2. Device responds with code HTTP 200 OK and CreatePullPointSubscriptionResponse message.
3. Client invokes PullMessages command with Timeout and MessageLimit elements.
4. Device responds with code HTTP 200 OK and PullMessagesResponse message.

Test Result:

PASS -

- Client **CreatePullPointSubscription** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **CreatePullPointSubscription** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<CreatePullPointSubscription>" tag after the "<Body>" tag AND
  - [S2] Device response contains "HTTP/* 200 OK" AND

- [S3] Device response contains "<CreatePullPointSubscriptionResponse>" tag AND

- Client **PullMessages** request messages are valid according to XML Schemas listed in Namespaces AND

- Client **PullMessages** request in Test Procedure fulfills the following requirements:

  - [S4] Client request contains "<PullMessages>" tag after the "<Body>" tag AND

  - [S7] Device response contains "HTTP/* 200 OK" AND

  - [S8] Device response contains "<PullMessagesResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: EventHandling.PullPoint

10.4 BASE NOTIFICATION

Test Label: Event Handling - Basic Notification

Test Case ID: EVENTHANDLING-2

Profile S Normative Reference: Conditional

Profile G Normative Reference: Conditional

Profile C Normative Reference: Governed by business rule #3

Profile Q Normative Reference: None

Profile A Normative Reference: None

Profile T Normative Reference: None

Feature Under Test: Event Handling

Test Purpose: To verify that the Client is able to retrieve events using WS-Base Notification.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with Basic Notification event type.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes Subscribe message with ConsumerReference element.
2. Device responds with code HTTP 200 OK and SubscribeResponse message.

Test Result:

PASS -

- Client **Subscribe** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **Subscribe** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<Subscribe>" tag after the "<Body>" tag AND
  - [S4] Device response contains "HTTP/* 200 OK" AND
  - [S5] Device response contains "<SubscribeResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: EventHandling.WSBaseNotification

10.5 METADATA STREAMING

Test Label: Event Handling - Metadata Streaming

Test Case ID: EVENTHANDLING-3

Profile S Normative Reference: Conditional

Profile G Normative Reference: None

Profile C Normative Reference: None

Profile Q Normative Reference: None

Profile A Normative Reference: None

Profile T Normative Reference: Conditional

Feature Under Test: Event Handling

Test Purpose: To verify that the Client is able to retrieve the Metadata Streaming.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with Metadata Streaming event type.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetStreamUri** request message for Media service or Media2 service for media profile that contains Video Source Configuration and Metadata Configuration. GetStreamUri request is set for RTP-Unicast/UDP OR RTP-Multicast/UDP OR RTP/RTSP/TCP OR RTP-Unicast/RTSP/HTTP/TCP transport.
2. Device responds with code HTTP 200 OK and **GetStreamUriResponse** message.
3. Client invokes **RTSP DESCRIBE** request to retrieve media stream description.
4. Device responds with code RTSP 200 OK and SDP information with Media Type: "application" and with encoding name "vnd.onvif.metadata" or "vnd.onvif.metadata.gzip" or "vnd.onvif.metadata.exi.onvif" or "vnd.onvif.metadata.exi.ext".
5. Client invokes **RTSP SETUP** request without "onvif-replay" Require header and with transport parameter element to to set media session parameters for metadata streaming.
6. Device responds with code RTSP 200 OK.
7. Client invokes **RTSP PLAY** request without "onvif-replay" Require header to start media stream.
8. Device responds with code RTSP 200 OK.
9. Client invokes **RTSP TEARDOWN** request to terminate the RTSP session.
10. If Device sends response to RTSP TEARDOWN, it has code RTSP 200 OK or RTSP 454.

Test Result:

Note: RTSP requests and RTSP response could be tunneled in HTTP if RTP-Unicast/RTSP/HTTP/TCP transport is used.

PASS -

- There is Client **RTSP DESCRIBE** request in Test Procedure
- Device response on the **RTSP DESCRIBE** request fulfills the following requirements:
  - [S1] It has RTSP 200 response code AND
  - [S2] SDP packet contains media type "application" (m=application) with sessions attribute "rtpmap" with encoding name "vnd.onvif.metadata" OR "vnd.onvif.metadata.gzip" OR

"vnd.onvif.metadata.exi.onvif" OR "vnd.onvif.metadata.exi.ext" (see ONVIF Streaming Spec) AND

- There is Client **RTSP SETUP** request in Test Procedure fulfills the following requirements:
  - [S3] It invoked for the same Device as for the Client **RTSP DESCRIBE** request AND
  - [S4] It invoked after the Client **RTSP DESCRIBE** request AND
  - [S5] RTSP address that was used to send **RTSP SETUP** is correspond to corresponding media Control URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
  - [S6] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP SETUP** request fulfills the following requirements:
  - [S7] It has RTSP 200 response code AND
- There is a Device response on the **GetStreamUri** request invoked EITHER for Media Service OR for Media2 Service in Test Procedure fulfills the following requirements:
  - [S8] It has HTTP 200 response code AND
  - [S9] It received for the same Device as for the Client **RTSP DESCRIBE** request AND
  - [S10] It received before the Client **RTSP DESCRIBE** request AND
  - [S11] In case Media Service, it contains **trt:MediaUri\tt:Uri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request. In case Media2 Service, it contains **tr2:GetStreamUriResponse\tr2:Uri** element which value is equal to RTSP address that was used to send the **RTSP DESCRIBE** request AND
- There is Client **RTSP PLAY** request in Test Procedure fulfills the following requirements:
  - [S12] It invoked for the same Device as for the Client **RTSP SETUP** request AND
  - [S13] It invoked after the Client **RTSP SETUP** request AND
  - [S14] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
  - [S15] It does not contain **Require** request header field with value is equal to "onvif-replay" AND
- Device response on the **RTSP PLAY** request fulfills the following requirements:

- [S16] It has RTSP 200 response code AND

- There is Client **RTSP TEARDOWN** request in Test Procedure fulfills the following requirements:
  - [S17] It invoked for the same Device as for the Client **RTSP SETUP** request AND
  - [S18] It invoked after the Client **RTSP PLAY** request AND
  - [S19] RTSP address that was used to send it is correspond to corresponding media Control URL or session Control URL or Content-Base URL from SDP packet (see [RFC 2326, C.1.1 Control URL]) AND
- If there is Device response on the **RTSP TEARDOWN** request then it fulfills the following requirements:
  - [S20] It has RTSP 200 response code.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: EventHandling.MetadataStreaming

11 Set Synchronization Point Test Cases

Validated Feature: SetSynchronizationPoint

Check Condition based on Device Features: None

Required Number of Devices: 1

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile S Requirement: Optional

Profile Q Requirement: Optional

Profile G Requirement: Optional

Profile T Normative Reference: Mandatory

11.1 Expected Scenarios Under Test:

1. Client connects to Device to synchronize property states.
2. Client is considered as supporting Set Synchronization Point if the following conditions are met:
   - Client is able to synchronize property states using **SetSynchronizationPoint** operation for subscribtions AND
3. Client is considered as NOT supporting Set Synchronization Point if the following is TRUE:
   - No valid responses for **SetSynchronizationPoint** request OR
   - **SetSynchronizationPoint** request does not contains valid **wsa:Action** header.

11.2 SET SYNCHRONIZATION POINT

Test Label: Set Synchronization Point - Set Synchronization Point

Test Case ID: SETSYNCHRONIZATIONPOINT-1

Profile A Normative Reference: Mandatory

Profile C Normative Reference: Mandatory

Profile S Normative Reference: Conditional

Profile Q Normative Reference: Optional

Profile G Normative Reference: Conditional

Profile T Normative Reference: Mandatory

Feature Under Test: SetSynchronizationPoint

Test Purpose: To verify that the Client is able to use **SetSynchronizationPoint** operation for subscribtion.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetSynchronizationPoint** operations present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetSynchronizationPoint** message with valid **wsa:Action** header to synchronize its properties with the properties of the device.
2. Device responses with code HTTP 200 OK and **SetSynchronizationPointResponse** message.

Test Result:

PASS -

- Client **SetSynchronizationPoint** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetSynchronizationPoint** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tev:SetSynchronizationPoint** AND
  - [S2] It contains **wsa:Action** element in header equal to "http://www.onvif.org/ver10/events/wsdl/PullPointSubscription/SetSynchronizationPointRequest" AND
- Device response on the **SetSynchronizationPoint** request fulfills the following requirements:
  - [S3] It has HTTP 200 response code AND
  - [S4] **soapenv:Body** element has child element **tev:SetSynchronizationPointResponse**

FAIL -

- The Client failed PASS criteria.

Validated Feature List: SetSynchronizationPoint.set_synchronization_point

12 Unsubscribe Test Cases

Validated Feature: Unsubscribe

Check Condition based on Device Features: None

Required Number of Devices: 1

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile S Requirement: Optional

Profile Q Requirement: Optional

Profile G Requirement: Optional

Profile T Requirement: Optional

12.1 Expected Scenarios Under Test:

1. Client connects to Device to Unsubscribe subscribtions.
2. Client is considered as supporting Unsubscribe if the following conditions are met:
   - Client is able to unsubscribe subscribtions using **Unsubscribe** operation.
3. Client is considered as NOT supporting Unsubscribe if the following is TRUE:
   - No valid responses for **Unsubscribe** request OR
   - **Unsubscribe** request does not contains valid **wsa:Action** header.

12.2 UNSUBSCRIBE

Test Label: Unsubscribe - Unsubscribe

Test Case ID: UNSUBSCRIBE-1

Profile A Requirement: Mandatory

Profile C Requirement: Mandatory

Profile S Requirement: Conditional

Profile Q Requirement: Optional

Profile G Requirement: Conditional

Profile T Requirement: Optional

Feature Under Test: Unsubscribe

Test Purpose: To verify that the Client is able to use **Unsubscribe** operation to terminate a subscribtion.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **Unsubscribe** operations present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **Unsubscribe** message with valid **wsa:Action** header to terminete a subscription.
2. Device responses with code HTTP 200 OK and **UnsubscribeResponse** message.

Test Result:

PASS -

- Client **Unsubscribe** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **Unsubscribe** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **wsnt:Unsubscribe** AND
  - [S2] It contains **wsa:Action** element in header equal to "http://docs.oasis-open.org/wsn/bw-2/SubscriptionManager/UnsubscribeRequest" AND
- Device response on the **Unsubscribe** request fulfills the following requirements:
  - [S3] It has HTTP 200 response code AND
  - [S4] **soapenv:Body** element has child element **wsnt:UnsubscribeResponse**

FAIL -

- The Client failed PASS criteria.

Validated Feature List: Unsubscribe.unsubscribe

13 Keep Alive for Pull Point Event Handling Test Cases

13.1 Feature Level Requirement:

Validated Feature: KeepAliveForPullPointEventHandling

Check Condition based on Device Features: None

Required Number of Devices: 3

Profile A Requirement: Mandatory

Profile C Requirement: Mandatory

Profile S Requirement: Conditional

Profile Q Requirement: Optional

Profile G Requirement: Conditional

Profile T Requirement: Optional

13.2 Expected Scenarios Under Test:

1. Client connects to Device to initiate Pull Point Event Handling.
2. Client is considered as supporting Keep Alive for Pull Point Event Handling if the following conditions are met:
   - Client supports EventHandling_Pullpoint feature AND
   - Client is able to renew pull point subscribtion using **Renew** operation OR **PullMessages** operation mechanism.
3. Client is considered as NOT supporting Keep Alive for Pull Point Event Handling if the following is TRUE:
   - No valid responses for **Renew** request AND for **CreatePullPointSubscription** request in the case if **PullMessages** used for keep alive OR
   - No valid responses for **Renew** request if detected OR
   - No valid responses for **CreatePullPointSubscription** request in the case if **PullMessages** used for keep alive if detected OR

- **Renew** request was invoked to address which was not specified in **tev:SubscriptionReference\wsa:Address** element of corresponding **CreatePullPointSubscriptionResponse** message.

13.3 RENEW

Test Label: Advanced Pull Point Event Handling - Renew

Test Case ID: KEEPALIVEFORPULLPOINTEVENTHANDLING-1

Profile A Normative Reference: Mandatory

Profile C Normative Reference: Mandatory

Profile S Normative Reference: Conditional

Profile Q Normative Reference: Optional

Profile G Normative Reference: Conditional

Profile T Normative Reference: Optional

Feature Under Test: Renew

Test Purpose: To verify that the Client is able to use **Renew** operation as keep alive for Pull Point subscribtion.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **Renew** operations present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **CreatePullPointSubscription** message.
2. Device responds with code HTTP 200 OK and **CreatePullPointSubscriptionResponse** message.
3. Client invokes **Renew** message to valid address recieved in **CreatePullPointSubscriptionResponse** message for the created Pull Point subscribtion with valid address recieved in **CreatePullPointSubscriptionResponse** message.
4. Device responds with code HTTP 200 OK and **RenewResponse** message.

Test Result:

PASS -

- Client **Renew** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **Renew** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **wsnt:Renew** AND
- Device response on the **Renew** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **wsnt:RenewResponse** AND
- There is a Device response on the **CreatePullPointSubscription** request in Test Procedure fulfills the following requirements:
  - [S4] It has HTTP 200 response code AND
  - [S5] It received for the same Device as for the Client **Renew** request AND
  - [S6] It received before the Client **Renew** request AND
  - [S7] It contains **tev:SubscriptionReference\wsa:Address** element which is equal to HTTP address that was used to send the **Renew** request.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: KeepAliveForPullPointEventHandling.Renew

13.4 PULL MESSAGES AS KEEP ALIVE

Test Label: Advanced Pull Point Event Handling - Pull Messages as Keep Alive

Test Case ID: KEEPALIVEFORPULLPOINTEVENTHANDLING-2

Profile A Requirement: Mandatory

Profile C Requirement: Mandatory

Profile S Requirement: Conditional

Profile Q Requirement: Optional

Profile G Requirement: Conditional

Profile T Requirement: Optional

Feature Under Test: Renew

Test Purpose: To verify that the Client is able to use **PullMessages** operation as keep alive for Pull Point subscribtion.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **CreatePullPointSubscription** operations whithout **tev:InitialTerminationTime** element present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **CreatePullPointSubscription** message.
2. Device responds with code HTTP 200 OK and **CreatePullPointSubscriptionResponse** message whithout **tev:InitialTerminationTime** element.

Test Result:

PASS -

- Client **CreatePullPointSubscription** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **CreatePullPointSubscription** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tev:CreatePullPointSubscription** AND
  - [S2] It does not contain **tev:InitialTerminationTime** element AND
- Device response on the **CreatePullPointSubscription** request fulfills the following requirements:
  - [S3] It has HTTP 200 response code AND
  - [S4] **soapenv:Body** element has child element **tev:CreatePullPointSubscriptionResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: KeepAliveForPullPointEventHandling.PullMessagesAsKeepAlive

14 Discovery Test Cases

14.1 Feature Level Requirement:

Validated Feature: Discovery

Check Condition based on Device Features: None

Required Number of Devices: 3

Profile S Requirement: Conditional

Profile C Requirement: Conditional

Profile G Requirement: Conditional

Profile A Requirement: Mandatory

Profile Q Requirement: Mandatory

Profile T Requirement: Mandatory

14.2 Expected Scenarios Under Test:

1. Client sends Probe message to multicast IP address 239.255.255.250 and port 3702 to locate services on a local network.
2. Client is considered as supporting Discovery if the following conditions are met:
   - Probe request detected AND at least one ProbeMatch response detected
3. Client is considered as NOT supporting Discovery if the following is TRUE:
   - No Valid Device Response to Probe request.

14.3 WS-DISCOVERY

Test Label: Discovery - WS-Discovery

Test Case ID: DISCOVERY-1

Profile S Normative Reference: Conditional

Profile G Normative Reference: Conditional

Profile C Normative Reference: Conditional

Profile Q Normative Reference: Mandatory

Profile A Normative Reference: Mandatory

Profile T Normative Reference: Mandatory

Feature Under Test: WS-Discovery

Test Purpose: To verify that Client is able to send Probe request and receive ProbeMatch response from Device.

Pre-Requisite:

- The Network Trace Capture files contain at least one Client Probe request to multicast IP address and one ProbeMatch response from Device directly to the Client.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes Probe request message to multicast IP address 239.255.255.250 and port 3702.
2. Device sends ProbeMatch message directly to the Client.

Test Result:

PASS -

- Client **Probe** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **Probe** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<Action>" tag after the "<Header>" tag AND
  - [S2] "<Action>" includes URL address which ends with "Probe" value AND
  - [S3] Client request contains "<MessageID>" with non-empty string value AND
  - [S4] Client request contains "<Probe>" tag after the "<Body>" tag AND
  - [S5] Device response message contains "<ProbeMatches>" tag after the "<Body>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: Discovery.WSDiscovery

15 Network Video Transmitter Discovery Type Filter Test Cases

15.1 Feature Level Requirement:

Validated Feature: NVTDiscoveryTypeFilter

Check Condition based on Device Features: Network Video Transmitter Discovery Type is supported by Device.

Required Number of Devices: 3

Profile S Requirement: Conditional

Profile A Requirement: None

Profile C Requirement: None

Profile Q Requirement: None

Profile G Requirement: None

Profile T Requirement: None

15.2 Expected Scenarios Under Test:

1. Client sends Probe message to multicast IPv4 address 239.255.255.250 or multicast IPv6 address [FF02::C] and port 3702 with Types filter is equal to **dn:NetworkVideoTransmitter** or with skipped Types filter.
2. Client is considered as supporting Network Video Transmitter Discovery Type if the following conditions are met:
   - **Probe** Client message that fulfills the following requirement is detected:
     - Types filter is equal to dn:NetworkVideoTransmitter or empty or skipped AND
     - Probe is sent to multicast IPv4 address 239.255.255.250 or multicast IPv6 address [FF02::C] AND
     - Probe is sent to UDP port 3702 AND
   - There is **ProbeMatch** Device message that correspond to Client **Probe**.

3. Client is considered as NOT supporting Network Video Transmitter Discovery Type if the following is TRUE:

   - No valid Device **ProbeMatch** message that is correspond to Client **Probe** message.

15.3 NVT DISCOVERY TYPE FILTER

Test Label: Discovery - Network Video Transmitter Discovery Type Filter

Test Case ID: NVTDISCOVERYTYPEFILTER-1

Profile S Normative Reference: Mandatory

Profile G Normative Reference: None

Profile C Normative Reference: None

Profile Q Normative Reference: None

Profile A Normative Reference: None

Profile T Normative Reference: None

Feature Under Test: WS-Discovery

Test Purpose: To verify that Client is able to discover devices with Network Video Transmitter Discovery Type.

Pre-Requisite:

- The Network Trace Capture files contains at least one Client Probe message that does not filter out devices with Network Video Transmitter Discovery Type that is sent to multicast WS-Discovery address.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes Probe request message to multicast IPv4 address 239.255.255.250 or multicast IPv6 address [FF02::C] and port 3702 with **Types** = dn:NetworkVideoTransmitter.
2. Device sends ProbeMatch message directly to the Client.

Test Result:

PASS -

- Client **Probe** request messages are valid according to XML Schemas listed in Namespaces AND

- Client **Probe** request in Test Procedure fulfills the following requirements:
  - [S1] It is sent to 239.255.255.250 IPv4 address OR [FF02::C] IPv6 address AND
  - [S2] It is sent to 3702 UDP port AND
  - [S3] **soapenv:Envelope/soapenv:Header** element has child element **wsadis:Action** AND
  - [S4] **wsadis:Action** includes URL address which ends with "Probe" value AND
  - [S5] **soapenv:Envelope/soapenv:Header** element has child element **wsadis:MessageID** with non-empty string value AND
  - [S6] **soapenv:Body** element has child element **d:Probe** AND
  - [S7] IF **d:Probe** element has child element **d:Types** THEN it has value is equal to **dn:NetworkVideoTransmitter** OR empty string value AND
  - [S8] There is Device **ProbeMatches** message in test procedure that fulfills the following requirements:
    - [S9] It is sent to Client IP address AND
    - [S10] **soapenv:Body** element has child element **d:ProbeMatches** AND
    - [S11] **soapenv:Envelope/soapenv:Header/wsadis:RelatesTo** element value is equeal to **soapenv:Envelope/soapenv:Header/wsadis:MessageID** value in **Probe** message AND

PASS WITH WARNING -

- **d:Probe/d:Types** element is skipped OR
- **d:Probe/d:Types** element has empty string value.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: NVTDiscoveryTypeFilter.NVTDiscoveryTypeFilter

16 Device Discovery Type Filter Test Cases

16.1 Feature Level Requirement:

Validated Feature: DeviceDiscoveryTypeFilter

Check Condition based on Device Features: Device Discovery Type is supported by Device.

Required Number of Devices: 3

Profile S Requirement: None

Profile A Requirement: Mandatory

Profile C Requirement: Conditional

Profile Q Requirement: Mandatory

Profile G Requirement: Conditional

Profile T Requirement: Mandatory

16.2 Expected Scenarios Under Test:

1. Client sends Probe message to multicast IPv4 address 239.255.255.250 or multicast IPv6 address [FF02::C] and port 3702 with Types filter is equal to **tds:Device** or with skipped Types filter.
2. Client is considered as supporting Device Discovery Type if the following conditions are met:
   - **Probe** Client message that fulfills the following requirement is detected:
     - Types filter is equal to tds:Device or empty or skipped AND
     - Probe is sent to multicast IPv4 address 239.255.255.250 or multicast IPv6 address [FF02::C] AND
     - Probe is sent to UDP port 3702 AND
   - There is **ProbeMatch** Device message that correspond to Client **Probe**.
3. Client is considered as NOT supporting Device Discovery Type if the following is TRUE:
   - No valid Device **ProbeMatch** message that is correspond to Client **Probe** message.

16.3 DEVICE DISCOVERY TYPE FILTER

Test Label: Discovery - Device Discovery Type Filter

Test Case ID: DEVICEDISCOVERYTYPEFILTER-1

Profile S Normative Reference: None

Profile G Normative Reference: Mandatory

Profile C Normative Reference: Mandatory

Profile Q Normative Reference: Mandatory

Profile A Normative Reference: Mandatory

Profile T Normative Reference: Mandatory

Feature Under Test: WS-Discovery

Test Purpose: To verify that Client is able to discover devices with Device Discovery Type.

Pre-Requisite:

- The Network Trace Capture files contains at least one Client Probe message that does not filter out devices with Device Discovery Type that is sent to multicast WS-Discovery address.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes Probe request message to multicast IPv4 address 239.255.255.250 or multicast IPv6 address [FF02::C] and port 3702 with **Types** = tds:Device.
2. Device sends ProbeMatch message directly to the Client.

Test Result:

PASS -

- Client **Probe** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **Probe** request in Test Procedure fulfills the following requirements:
  - [S1] It is sent to 239.255.255.250 IPv4 address OR [FF02::C] IPv6 address AND
  - [S2] It is sent to 3702 UDP port AND
  - [S3] **soapenv:Envelope/soapenv:Header** element has child element **wsadis:Action** AND

- [S4] **wsadis:Action** includes URL address which ends with "Probe" value AND
- [S5] **soapenv:Envelope/soapenv:Header** element has child element **wsadis:MessageID** with non-empty string value AND
- [S6] **soapenv:Body** element has child element **d:Probe** AND
- [S7] IF **d:Probe** element has child element **d:Types** THEN it has value is equal to **tds:Device** OR empty string value AND
- [S8] There is Device **ProbeMatches** message in test procedure that fulfills the following requirements:
  - [S9] It is sent to Client IP address AND
  - [S10] **soapenv:Body** element has child element **d:ProbeMatches** AND
  - [S11] **soapenv:Envelope/soapenv:Header/wsadis:RelatesTo** element value is equeal to **soapenv:Envelope/soapenv:Header/wsadis:MessageID** value in **Probe** message AND

PASS WITH WARNING -

- **d:Probe/d:Types** element is skipped OR
- **d:Probe/d:Types** element has empty string value.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: DeviceDiscoveryTypeFilter.DeviceDiscoveryTypeFilter

17 Network Configuration Test Cases

17.1 Feature Level Requirement:

Validated Feature: NetworkConfiguration

Check Condition based on Device Features: None

Required Number of Devices: 3

Profile A Requirement: Conditional

Profile C Requirement: Conditional

Profile G Requirement: Conditional

Profile Q Requirement: Conditional

Profile S Requirement: Conditional

Profile T Requirement: Mandatory

17.2 Expected Scenarios Under Test:

1. Client connects to Device to configure network settings.
2. Client is considered as supporting Network Configuration if the following conditions are met:
   - Client is able to list network interfaces of Device using the GetNetworkInterfaces operation AND
   - Client is able to set network interfaces of Device using the SetNetworkInterfaces operation AND
   - Client is able to list default gateway of Device using the GetNetworkDefaultGateway operation AND
   - Client is able set default gateway of Device using the SetNetworkDefaultGateway operation.
3. Client is considered as NOT supporting Network Configuration if ANY of the following is TRUE:
   - No Valid Device Response to GetNetworkInterfaces request OR
   - No Valid Device Response to SetNetworkInterfaces request OR

- No Valid Device Response to GetNetworkDefaultGateway request OR
- No Valid Device Response to SetNetworkDefaultGateway request.

17.3 GET NETWORK INTERFACES

Test Label: Network Configuration - Get Network Interfaces

Test Case ID: NETWORKCONFIGURATION-1

Profile S Normative Reference: Conditional

Profile G Normative Reference: Conditional

Profile C Normative Reference: Conditional

Profile Q Normative Reference: Conditional

Profile A Normative Reference: Conditional

Profile T Normative Reference: Mandatory

Feature Under Test: NetworkConfiguration

Test Purpose: To verify that Client is able to list network interfaces of Device using the GetNetworkInterfaces operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetNetworkInterfaces operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetNetworkInterfaces request message to get network interface configuration from Device.
2. Device responds with code HTTP 200 OK and GetNetworkInterfacesResponse message.

Test Result:

PASS -

- Client **GetNetworkInterfaces** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetNetworkInterfaces** request in Test Procedure fulfills the following requirements:

- [S1] Client request contains "<GetNetworkInterfaces>" tag after the "<Body>" tag AND
- [S2] Device response contains "HTTP/* 200 OK" AND
- [S3] Device response contains "<GetNetworkInterfacesResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: NetworkConfiguration.GetNetworkInterfaces

17.4 SET NETWORK INTERFACES

Test Label: Network Configuration - Set Network Interfaces

Test Case ID: NETWORKCONFIGURATION-2

Profile S Normative Reference: Conditional

Profile G Normative Reference: Conditional

Profile C Normative Reference: Conditional

Profile Q Normative Reference: Conditional

Profile A Normative Reference: Conditional

Profile T Normative Reference: Mandatory

Feature Under Test: NetworkConfiguration

Test Purpose: To verify that Client is able to set network interfaces of Device using the SetNetworkInterfaces operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetNetworkInterfaces operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetNetworkInterfaces request message to set the network interface configuration on Device.
2. Device responds with code HTTP 200 OK and SetNetworkInterfacesResponse message.

Test Result:

PASS -

- Client **SetNetworkInterfaces** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetNetworkInterfaces** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<SetNetworkInterfaces>" tag after the "<Body>" tag AND
  - [S2] "<SetNetworkInterfaces>" includes tag: "<InterfaceToken>" with non-empty string value of specific token AND
  - [S4] Device response contains "HTTP/* 200 OK" AND
  - [S5] Device response contains "<SetNetworkInterfacesResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: NetworkConfiguration.SetNetworkInterfaces

17.5 GET NETWORK DEFAULT GATEWAY

Test Label: Network Configuration - Get Network Default Gateway

Test Case ID: NETWORKCONFIGURATION-3

Profile S Normative Reference: Conditional

Profile G Normative Reference: Conditional

Profile C Normative Reference: Conditional

Profile Q Normative Reference: Conditional

Profile A Normative Reference: Conditional

Profile T Normative Reference: Mandatory

Feature Under Test: NetworkConfiguration

Test Purpose: To verify that Client is able to list default gateway of Device using the GetNetworkDefaultGateway operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetNetworkDefaultGateway operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetNetworkDefaultGateway request message to get the default gateway settings from Device.
2. Device responds with code HTTP 200 OK and GetNetworkDefaultGatewayResponse message.

Test Result:

PASS -

- Client **GetNetworkDefaultGateway** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetNetworkDefaultGateway** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<GetNetworkDefaultGateway>" tag after the "<Body>" tag AND
  - [S2] Device response contains "HTTP/* 200 OK" AND
  - [S3] Device response contains "<GetNetworkDefaultGatewayResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: NetworkConfiguration.GetNetworkDefaultGateway

17.6 SET NETWORK DEFAULT GATEWAY

Test Label: Network Configuration - Set Network Default Gateway

Test Case ID: NETWORKCONFIGURATION-4

Profile S Normative Reference: Conditional

Profile G Normative Reference: Conditional

Profile C Normative Reference: Conditional

Profile Q Normative Reference: Conditional

Profile A Normative Reference: Conditional

Profile T Normative Reference: Mandatory

Feature Under Test: NetworkConfiguration

Test Purpose: To verify that Client is able to set default gateway of Device using the SetNetworkDefaultGateway operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetNetworkDefaultGateway operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetNetworkDefaultGateway request message to set the default gateway settings on Device.
2. Device responds with code HTTP 200 OK and SetNetworkDefaultGatewayResponse message.

Test Result:

PASS -

- Client **SetNetworkDefaultGateway** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetNetworkDefaultGateway** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<SetNetworkDefaultGateway>" tag after the "<Body>" tag AND
  - [S2] "<SetNetworkDefaultGateway>" includes tag: EITHER "<IPv4Address>" OR "<IPv6Address>" with specific IP address value AND
  - [S3] Device response contains "HTTP/* 200 OK" AND
  - [S4] Device response contains "<SetNetworkDefaultGatewayResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: NetworkConfiguration.SetNetworkDefaultGateway

18 System Test Cases

18.1 Feature Level Requirement:

Validated Feature: System

Check Condition based on Device Features: None

Required Number of Devices: 3

Profile A Requirement: Conditional

Profile C Requirement: Conditional

Profile G Requirement: Conditional

Profile Q Requirement: Conditional

Profile S Requirement: Conditional

Profile T Requirement: Conditional

18.2 Expected Scenarios Under Test:

1. Client connects to Device to get information, such as manufacturer, model, firmware version and etc.
2. Client is considered as supporting System if the following conditions are met:
   - Client is able to list Device information using the GetDeviceInformation operation.
3. Client is considered as NOT supporting System if ANY of the following is TRUE:
   - No Valid Device Response to GetDeviceInformation request.

18.3 GET DEVICE INFORMATION

Test Label: System - Get Device Information

Test Case ID: SYSTEM-1

Profile S Normative Reference: Conditional

Profile G Normative Reference: Conditional

Profile C Normative Reference: Conditional

Profile Q Normative Reference: Conditional

Profile A Normative Reference: Conditional

Profile T Normative Reference: Conditional

Feature Under Test: System

Test Purpose: To verify that Client is able to list Device information using the GetDeviceInformation operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetDeviceInformation operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetDeviceInformation request message to list Device information.
2. Device responds with code HTTP 200 OK and GetDeviceInformationResponse message.

Test Result:

PASS -

- Client **GetDeviceInformation** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetDeviceInformation** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<GetDeviceInformation>" tag after the "<Body>" tag AND
  - [S2] Device response contains "HTTP/* 200 OK" AND
  - [S3] Device response contains "<GetDeviceInformationResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: System.GetDeviceInformation

19 User Handling Test Cases

19.1 Feature Level Requirement:

Validated Feature: UserHandling

Check Condition based on Device Features: None

Required Number of Devices: 3

Profile A Requirement: Mandatory

Profile Q Requirement: Mandatory

Profile S Requirement: Conditional

Profile C Requirement: Conditional

Profile G Requirement: Conditional

Profile T Requirement: Conditional

19.2 Expected Scenarios Under Test:

1. Client connects to Device to create, list, modify and delete users.
2. Client is considered as supporting User Handling if the following conditions are met:
   - Client is able to create users on Device using the CreateUsers operation AND
   - Client is able to list existing users of Device using the GetUsers operation AND
   - Client is able to modify users on Device using the SetUser operation AND
   - Client is able to delete users from Device using the DeleteUsers operation.
3. Client is considered as NOT supporting System if ANY of the following is TRUE:
   - No Valid Device Response to CreateUsers request (except SOAP fault: **soapenv:Receiver/ter:Action/ter:TooManyUsers**) OR
   - No Valid Device Response to GetUsers request OR
   - No Valid Device Response to SetUser request (except SOAP fault: **soapenv:Sender/ter:InvalidArgVal/ter:FixedUser**) OR

- No Valid Device Response to DeleteUsers request (except SOAP fault: **soapenv:Sender/ter:InvalidArgVal/ter:FixedUser**).

19.3 CREATE USERS

Test Label: User Handling - CreateUsers

Test Case ID: USERHANDLING-1

Profile S Normative Reference: Conditional

Profile G Normative Reference: Conditional

Profile C Normative Reference: Conditional

Profile Q Normative Reference: Mandatory

Profile A Normative Reference: Mandatory

Profile T Normative Reference: Conditional

Feature Under Test: User Handling

Test Purpose: To verify that Client is able to create users on Device using the CreateUsers operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with CreateUsers operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes CreateUsers request message to create new users and corresponding credentials on Device.
2. Device responds with code HTTP 200 OK and CreateUsersResponse message.

Test Result:

PASS -

- Client **CreateUsers** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **CreateUsers** request in Test Procedure fulfills the following requirements:

- [S1] Client request contains "<CreateUsers>" tag after the "<Body>" tag AND
- [S2] "<CreateUsers>" includes tag: "<User>" AND
- [S3] "<User>" includes tag: "<Username>" with non-empty string value AND
- [S4] "<User>" includes tag: "<Password>" with non-empty string value AND
- [S5] If Device response contains "HTTP/* 200 OK" THEN it contains "<CreateUsersResponse>" tag, ELSE it contains **soapenv:Fault** with **soapenv:Receiver/ter:Action/ter:TooManyUsers** fault code.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: UserHandling.CreateUsers

19.4 GET USERS

Test Label: User Handling - GetUsers

Test Case ID: USERHANDLING-2

Profile S Normative Reference: Conditional

Profile G Normative Reference: Conditional

Profile C Normative Reference: Conditional

Profile Q Normative Reference: Mandatory

Profile A Normative Reference: Mandatory

Profile T Normative Reference: Conditional

Feature Under Test: User Handling

Test Purpose: To verify that Client is able to list existing users of Device using the GetUsers operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetUsers operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetUsers request message to list registered users and their user levels.
2. Device responds with code HTTP 200 OK and GetUsersResponse message.

Test Result:

PASS -

- Client **GetUsers** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetUsers** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<GetUsers>" tag after the "<Body>" tag AND
  - [S2] Device response contains "HTTP/* 200 OK" AND
  - [S3] Device response contains "<GetUsersResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: UserHandling.GetUsers

19.5 SET USER

Test Label: User Handling - SetUser

Test Case ID: USERHANDLING-3

Profile S Normative Reference: Conditional

Profile G Normative Reference: Conditional

Profile C Normative Reference: Conditional

Profile Q Normative Reference: Mandatory

Profile A Normative Reference: Mandatory

Profile T Normative Reference: Conditional

Feature Under Test: User Handling

Test Purpose: To verify that Client is able to modify users on Device using the SetUser operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetUser operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetUser request message to update the authentication settings on Device.
2. Device responds with code HTTP 200 OK and SetUserResponse message.

Test Result:

PASS -

- Client **SetUser** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetUser** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<SetUser>" tag after the "<Body>" tag AND
  - [S2] "<SetUser>" includes tag: "<User>" AND
  - [S3] "<User>" includes tag: "<Username>" with non-empty string value AND
  - [S4] If Device response contains "HTTP/* 200 OK" THEN it contains "<SetUserResponse>" tag, ELSE it contains **soapenv:Fault** with **soapenv:Sender/ter:InvalidArgVal/ter:FixedUser** fault code.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: UserHandling.SetUser

19.6 DELETE USERS

Test Label: User Handling - DeleteUsers

Test Case ID: USERHANDLING-4

Profile S Normative Reference: Conditional

Profile G Normative Reference: Conditional

Profile C Normative Reference: Conditional

Profile Q Normative Reference: Mandatory

Profile A Normative Reference: Mandatory

Profile T Normative Reference: Conditional

Feature Under Test: User Handling

Test Purpose: To verify that Client is able to delete users from Device using the DeleteUsers operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with DeleteUsers operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes DeleteUsers request message to delete specific users from Device.
2. Device responds with code HTTP 200 OK and DeleteUsersResponse message.

Test Result:

PASS -

- Client **DeleteUsers** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **DeleteUsers** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<DeleteUsers>" tag after the "<Body>" tag AND
  - [S2] "<DeleteUsers>" includes tag: "<Username>" with non-empty string value AND
  - [S3] If Device response contains "HTTP/* 200 OK" THEN it contains "<DeleteUsersResponse>" tag, ELSE it contains **soapenv:Fault** with **soapenv:Sender/ter:InvalidArgVal/ter:FixedUser** fault code.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: UserHandling.DeleteUsers

20 Relay Outputs Test Cases

20.1 Feature Level Requirement:

Validated Feature: RelayOutputs

Check Condition based on Device Features: Relay Outputs (Device Management Service) is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

20.2 Expected Scenarios Under Test:

1. Client connects to Device to list, configure and trigger relay outputs using Device Management service.
2. Client is considered as supporting Relay Outputs if the following conditions are met:
   - Client is able to list available relay outputs using the GetRelayOutputs operation using Device Management service AND
   - Client is able to trigger relay output using the SetRelayOutputState operation using Device Management service AND
   - Client is able to set settings of relay output in EITHER "Bistable" OR "Monostable" mode using the SetRelayOutputSettings operation using Device Management service.
3. Client is considered as NOT supporting Relay Outputs if ANY of the following is TRUE:
   - No Valid Device Response to GetRelayOutputs request to Device Management service OR
   - No Valid Device Response to SetRelayOutputState request to Device Management service OR
   - No Valid Device Response to SetRelayOutputSettings requests to Device Management service for BOTH "Bistable" AND "Monostable" mode.

20.3 GET RELAY OUTPUTS

Test Label: Relay Output - Get Relay Outputs

Test Case ID: RELAYOUTPUTS-1

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Feature Under Test: Relay Outputs

Test Purpose: To verify that Client is able to list available relay outputs using the GetRelayOutputs operation for Device Management Service.

Test Purpose: To verify that relay outputs provided by Device is received by Client using the **GetRelayOutputs** operation using Device Management Service.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetRelayOutputs** operation for Device Management Service present.
- Client supports Capabilities feature.
- The Client Test Tool retrives Device Management Service address from device's response on GetServices or GetCapabilities Client request.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetRelayOutputs** request message to Device Management Service to retrieve relay outputs from the Device.
2. Device responds with code HTTP 200 OK and **GetRelayOutputsResponse** message.

Test Result:

PASS -

- Client **GetRelayOutputs** request messages to Device Management Service are valid according to XML Schemas listed in Namespaces AND
- Client **GetRelayOutputs** request to Device Management Service in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:GetRelayOutputs** AND
- Device response on the **GetRelayOutputs** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND

- [S3] **soapenv:Body** element has child element **tds:GetRelayOutputsResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: RelayOutputs.GetRelayOutputs

20.4 SET RELAY OUTPUT STATE

Test Label: Relay Output - Set Relay Output State

Test Case ID: RELAYOUTPUTS-2

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Feature Under Test: Relay Outputs

Test Purpose: To verify that Client is able to trigger a relay output using the **SetRelayOutputState** operation for Device Management Service.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetRelayOutputState** operation for Device Management Service present.
- Client supports Capabilities feature.
- The Client Test Tool retrives Device Management Service address from device's response on GetServices or GetCapabilities Client request.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetRelayOutputState** request message to Device Management Service to trigger a relay output on the Device.
2. Device responds with code HTTP 200 OK and **SetRelayOutputStateResponse** message.

Test Result:

PASS -

- Client **SetRelayOutputState** request messages to Device Management Service are valid according to XML Schemas listed in Namespaces AND
- Client **SetRelayOutputState** request to Device Management Service in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SetRelayOutputState** AND
  - [S2] **tds:SetRelayOutputState\tds:RelayOutputToken** element has non-empty string value AND
- Device response on the **SetRelayOutputState** request fulfills the following requirements:
  - [S3] It has HTTP 200 response code AND
  - [S4] **soapenv:Body** element has child element **tds:SetRelayOutputStateResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: RelayOutputs.SetRelayOutputState

20.5 SET RELAY OUTPUT SETTINGS BISTABLE MODE

Test Label: Relay Outputs - SetRelayOutputSettings Bistable Mode

Test Case ID: RELAYOUTPUTS-3

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Feature Under Test: Relay Outputs

Test Purpose: To verify that Client is able to set settings of relay output in "Bistable" mode using the **SetRelayOutputSettings** operation for Device Management Service.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetRelayOutputSettings** operation for Device Management Service with **tds:SetRelayOutputSettings\tds:Properties\tt:Mode** element value is equal to "Bistable" present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetRelayOutputSettings** request message to Device Management Service to set setting of relay output in "Bistable" mode.
2. Device responds with code HTTP 200 OK and **SetRelayOutputSettingsResponse** message.

Test Result:

NOTE: If Client SetRelayOutputSettings request message does not contain "Bistable" value of Mode element then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **SetRelayOutputSettings** request messages to Device Management Service are valid according to XML Schemas listed in Namespaces AND
- Client **SetRelayOutputSettings** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SetRelayOutputSettings** AND
  - [S2] **tds:SetRelayOutputSettings\tds:RelayOutputToken** element has non-empty string value AND
  - [S2] **tds:SetRelayOutputSettings\tds:Properties\tt:Mode** element value is equal to "Bistable" AND
- Device response on the **SetRelayOutputSettings** request fulfills the following requirements:
  - [S4] It has HTTP 200 response code AND
  - [S5] **soapenv:Body** element has child element **tds:SetRelayOutputSettingsResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: RelayOutputs.SetRelayOutputBistable

20.6 SET RELAY OUTPUT SETTINGS MONOSTABLE MODE

Test Label: Relay Outputs - SetRelayOutputSettings Monostable Mode

Test Case ID: RELAYOUTPUTS-4

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Feature Under Test: Relay Outputs

Test Purpose: To verify that Client is able to set settings of relay output in "Monostable" mode using the **SetRelayOutputSettings** operation for Device Management Service.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetRelayOutputSettings** operation for Device Management Service with **tds:SetRelayOutputSettings\tds:Properties\tt:Mode** element value is equal to "Monostable" present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetRelayOutputSettings** request message to Device Management Service to set setting of relay output in "Monostable" mode.
2. Device responds with code HTTP 200 OK and **SetRelayOutputSettingsResponse** message.

Test Result:

NOTE: If Client SetRelayOutputSettings request message does not contain "Monostable" value of Mode element then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **SetRelayOutputSettings** request messages to Device Management Service are valid according to XML Schemas listed in Namespaces AND
- Client **SetRelayOutputSettings** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SetRelayOutputSettings** AND
  - [S2] **tds:SetRelayOutputSettings\tds:RelayOutputToken** element has non-empty string value AND
  - [S2] **tds:SetRelayOutputSettings\tds:Properties\tt:Mode** element value is equal to "Monostable" AND
- Device response on the **SetRelayOutputSettings** request fulfills the following requirements:
  - [S4] It has HTTP 200 response code AND

- [S5] **soapenv:Body** element has child element **tds:SetRelayOutputSettingsResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: RelayOutputs.SetRelayOutputMonostable

21 NTP Test Cases

21.1 Feature Level Requirement:

Validated Feature: NTP

Check Condition based on Device Features: NTP is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

Profile Q Requirement: Conditional

Profile T Requirement: Conditional

21.2 Expected Scenarios Under Test:

1. Client connects to Device to configure synchronization of time using NTP servers on Device.
2. Client is considered as supporting NTP if the following conditions are met:
   - Client is able to get the NTP settings from Device using the GetNTP operation AND
   - Client is able to set the NTP settings on Device using the SetNTP operation.
3. Client is considered as NOT supporting NTP if ANY of the following is TRUE:
   - No Valid Device Response to GetNTP request OR
   - No Valid Device Response to SetNTP request.

21.3 GET NTP

Test Label: NTP - GetNTP

Test Case ID: NTP-1

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Profile Q Normative Reference: Conditional

Profile T Normative Reference: Conditional

Feature Under Test: NTP

Test Purpose: To verify that Client is able to get the NTP settings from Device using the GetNTP operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetNTP operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetNTP request message to get current settings of NTP servers on Device.
2. Device responds with code HTTP 200 OK and GetNTPResponse message.

Test Result:

PASS -

- Client **GetNTP** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetNTP** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<GetNTP>" tag after the "<Body>" tag AND
  - [S2] Device response contains "HTTP/* 200 OK" AND
  - [S3] Device response contains "<GetNTPResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: NTP.GetNTP

21.4 SET NTP

Test Label: NTP - SetNTP

Test Case ID: NTP-2

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Profile Q Normative Reference: Conditional

Profile T Normative Reference: Conditional

Feature Under Test: NTP

Test Purpose: To verify that Client is able to set the NTP settings on Device using the SetNTP operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetNTP operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetNTP request message to set the NTP servers settings on Device.
2. Device responds with code HTTP 200 OK and SetNTPResponse message.

Test Result:

PASS -

- Client **SetNTP** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetNTP** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<SetNTP>" tag after the "<Body>" tag AND
  - [S2] Device response contains "HTTP/* 200 OK" AND
  - [S3] Device response contains "<SetNTPResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: NTP.SetNTP

22 Dynamic DNS Test Cases

22.1 Feature Level Requirement:

Validated Feature: DynamicDns

Check Condition based on Device Features: Dynamic DNS is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

22.2 Expected Scenarios Under Test:

1. Client connects to Device to configure Dynamic DNS settings.
2. Client is considered as supporting Dynamic DNS if the following conditions are met:
   - Client is able to get the Dynamic DNS settings from Device using the GetDynamicDNS operation AND
   - Client is able to set the Dynamic DNS settings on Device using the SetDynamicDNS operation.
3. Client is considered as NOT supporting Dynamic DNS if ANY of the following is TRUE:
   - No Valid Device Response to GetDynamicDNS request OR
   - No Valid Device Response to SetDynamicDNS request.

22.3 GET DYNAMIC DNS SETTINGS

Test Label: Dynamic DNS - GetDynamicDNS

Test Case ID: DYNAMICDNS-1

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Feature Under Test: Dynamic DNS

Test Purpose: To verify that Client is able get the dynamic DNS settings from Device using the GetDynamicDNS operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetDynamicDNS operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetDynamicDNS request message to get the dynamic DNS settings from Device.
2. Device responds with code HTTP 200 OK and GetDynamicDNSResponse message.

Test Result:

PASS -

- Client **GetDynamicDNS** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetDynamicDNS** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<GetDynamicDNS>" tag after the "<Body>" tag AND
  - [S2] Device response contains "HTTP/* 200 OK" AND
  - [S3] Device response contains "<GetDynamicDNSResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: DynamicDns.GetDynamicDnsSettings

22.4 SET DYNAMIC DNS SETTINGS

Test Label: Dynamic DNS - SetDynamicDNS

Test Case ID: DYNAMICDNS-2

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Feature Under Test: Dynamic DNS

Test Purpose: To verify that Client is able set the dynamic DNS settings on Device using the SetDynamicDNS operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetDynamicDNS operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetDynamicDNS request message to set the dynamic DNS settings on Device.
2. Device responds with code HTTP 200 OK and SetDynamicDNSResponse message.

Test Result:

PASS -

- Client **SetDynamicDNS** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetDynamicDNS** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<SetDynamicDNS>" tag after the "<Body>" tag AND
  - [S3] Device response contains "HTTP/* 200 OK" AND
  - [S4] Device response contains "<SetDynamicDNSResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: DynamicDns.SetDynamicDnsSettings

23 Zero Configuration Test Cases

23.1 Feature Level Requirement:

Validated Feature: ZeroConfiguration

Check Condition based on Device Features: Zero Configuration is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

Profile Q Requirement: Conditional

23.2 Expected Scenarios Under Test:

1. Client connects to Device to configure Zero Configuration settings.
2. Client is considered as supporting Zero Configuration if the following conditions are met:
   - Client is able to get the Zero Configuration settings from Device using the GetZeroConfiguration operation AND
   - Client is able to set the Zero Configuration settings on Device using the SetZeroConfiguration operation.
3. Client is considered as NOT supporting Zero Configuration if ANY of the following is TRUE:
   - No Valid Device Response to GetZeroConfiguration request OR
   - No Valid Device Response to SetZeroConfiguration request.

23.3 GET ZERO CONFIGURATION

Test Label: Zero Configuration - GetZeroConfiguration

Test Case ID: ZEROCONFIGURATION-1

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Profile Q Normative Reference: Conditional

Feature Under Test: Zero Configuration

Test Purpose: To verify that Client is able to get the Zero Configuration settings from Device using the GetZeroConfiguration operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetZeroConfiguration operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetZeroConfiguration request message to get the Zero Configuration settings from Device.
2. Device responds with code HTTP 200 OK and GetZeroConfigurationResponse message.

Test Result:

PASS -

- Client **GetZeroConfiguration** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetZeroConfiguration** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<GetZeroConfiguration>" tag after the "<Body>" tag AND
  - [S2] Device response contains "HTTP/* 200 OK" AND
  - [S3] Device response contains "<GetZeroConfigurationResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: ZeroConfiguration.GetZeroConfiguration

23.4 SET ZERO CONFIGURATION

Test Label: Zero Configuration - SetZeroConfiguration

Test Case ID: ZEROCONFIGURATION-2

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Profile Q Normative Reference: Conditional

Feature Under Test: Zero Configuration

Test Purpose: To verify that Client is able to set the Zero Configuration settings on Device using the SetZeroConfiguration operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetZeroConfiguration operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetZeroConfiguration request message to set the Zero Configuration settings on Device.
2. Device responds with code HTTP 200 OK and SetZeroConfigurationResponse message.

Test Result:

PASS -

- Client **SetZeroConfiguration** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetZeroConfiguration** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<SetZeroConfiguration>" tag after the "<Body>" tag AND
  - [S2] "<SetZeroConfiguration>" includes tag: "<InterfaceToken>" with non-empty string value of specific token AND
  - [S3] Device response contains "HTTP/* 200 OK" AND
  - [S4] Device response contains "<SetZeroConfigurationResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: ZeroConfiguration.SetZeroConfiguration

24 IP Address Filtering Test Cases

24.1 Feature Level Requirement:

Validated Feature: IPAddressFiltering

Check Condition based on Device Features: IP Filter is supported by Device.

Required Number of Devices: 1

Profile S Requirement: Conditional

Profile C Requirement: Conditional

Profile A Requirement: Conditional

24.2 Expected Scenarios Under Test:

1. Client connects to Device to manage IP address filters.
2. Client is considered as supporting IP Address Filtering if the following conditions are met:
   - Client is able to get the IP address filter settings from Device using the GetIPAddressFilter operation AND
   - Client is able to set the IP address filter settings on Device using the SetIPAddressFilter operation AND
   - Client is able to add the IP address filter settings to Device using the AddIPAddressFilter operation AND
   - Client is able to delete the IP address filter settings from Device using the RemoveIPAddressFilter operation.
   - **NOTE:** Requests SetIPAddressFilter, AddIPAddressFilter and RemoveIPAddressFilter are permitted to use the IPv4 OR IPv6 protocol settings.
3. Client is considered as NOT supporting IP Address Filtering if ANY of the following is TRUE:
   - No Valid Device Response to GetIPAddressFilter request OR
   - No Valid Device Response to SetIPAddressFilter request OR
   - No Valid Device Response to AddIPAddressFilter request OR
   - No Valid Device Response to RemoveIPAddressFilter request.

- NOTE: Requests SetIPAddressFilter, AddIPAddressFilter and RemoveIPAddressFilter should be deemed as failed if both IPv4 AND IPv6 protocol settings have No Valid Device Responses.

24.3 GET IP ADDRESS FILTER

Test Label: IP Address Filtering - GetIPAddressFilter

Test Case ID: IPADDRESSFILTERING-1

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Conditional

Profile A Normative Reference: Conditional

Feature Under Test: IP Address Filtering

Test Purpose: To verify that Client is able to get the IP address filter settings from Device using the GetIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with GetIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes GetIPAddressFilter request message to get the IP address filter settings from Device.
2. Device responds with code HTTP 200 OK and GetIPAddressFilterResponse message.

Test Result:

PASS -

- Client **GetIPAddressFilter** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetIPAddressFilter** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<GetIPAddressFilter>" tag after the "<Body>" tag AND
  - [S2] Device response contains "HTTP/* 200 OK" AND

- [S3] Device response contains "<GetIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: IPAddressFiltering.GetIpAddressFilter

24.4 SET IPv4 ADDRESS FILTER

Test Label: IP Address Filtering - SetIPv4AddressFilter

Test Case ID: IPADDRESSFILTERING-2

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Conditional

Profile A Normative Reference: Conditional

Feature Under Test: IP Address Filtering

Test Purpose: To verify that Client is able to set the IP address filter settings on Device using the SetIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetIPAddressFilter request message to set the IP address filter settings on Device.
2. Device responds with code HTTP 200 OK and SetIPAddressFilterResponse message.

Test Result:

NOTE: If Client SetIPAddressFilter request message does not contain "<IPv4Address>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **SetIPAddressFilter** request messages are valid according to XML Schemas listed in Namespaces AND

- Client **SetIPAddressFilter** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<SetIPAddressFilter>" tag after the "<Body>" tag AND
  - [S3] "<SetIPAddressFilter>" includes tag: "<IPv4Address>" AND
  - [S4] "<IPv4Address>" includes tag: "<Address>" with specific IPv4 address value AND
  - [S5] "<IPv4Address>" includes tag: "<PrefixLength>" with value range from "0" to "32" AND
  - [S6] Device response contains "HTTP/* 200 OK" AND
  - [S7] Device response contains "<SetIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: IPAddressFiltering.SetIpV4AddressFilter

24.5 SET IPv6 ADDRESS FILTER

Test Label: IP Address Filtering - SetIPv6AddressFilter

Test Case ID: IPADDRESSFILTERING-3

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Conditional

Profile A Normative Reference: Conditional

Feature Under Test: IP Address Filtering

Test Purpose: To verify that Client is able to set the IP address filter settings on Device using the SetIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with SetIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes SetIPAddressFilter request message to set the IP address filter settings on Device.

2. Device responds with code HTTP 200 OK and SetIPAddressFilterResponse message.

Test Result:

NOTE: If Client SetIPAddressFilter request message does not contain "<IPv6Address>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **SetIPAddressFilter** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetIPAddressFilter** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<SetIPAddressFilter>" tag after the "<Body>" tag AND
  - [S3] "<SetIPAddressFilter>" includes tag: "<IPv6Address>" AND
  - [S4] "<IPv6Address>" includes tag: "<Address>" with specific IPv6 address value AND
  - [S5] "<IPv6Address>" includes tag: "<PrefixLength>" with value range from "0" to "128" AND
  - [S6] Device response contains "HTTP/* 200 OK" AND
  - [S7] Device response contains "<SetIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: IPAddressFiltering.SetIpV6AddressFilter

24.6 ADD IPv4 ADDRESS FILTER

Test Label: IP Address Filtering - AddIPv4AddressFilter

Test Case ID: IPADDRESSFILTERING-4

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Conditional

Profile A Normative Reference: Conditional

Feature Under Test: IP Address Filtering

Test Purpose: To verify that Client is able to add the IP address filter to Device using the AddIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with AddIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes AddIPAddressFilter request message to add the IP address filter on Device.
2. Device responds with code HTTP 200 OK and AddIPAddressFilterResponse message.

Test Result:

NOTE: If Client AddIPAddressFilter request message does not contain "<IPv4Address>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **AddIPAddressFilter** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **AddIPAddressFilter** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<AddIPAddressFilter>" tag after the "<Body>" tag AND
  - [S3] "<AddIPAddressFilter>" includes tag: "<IPv4Address>" AND
  - [S4] "<IPv4Address>" includes tag: "<Address>" with specific IPv4 address value AND
  - [S5] "<IPv4Address>" includes tag: "<PrefixLength>" with value range from "0" to "32" AND
  - [S6] Device response contains "HTTP/* 200 OK" AND
  - [S7] Device response contains "<AddIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: IPAddressFiltering.AddIpV4AddressFilter

24.7 ADD IPv6 ADDRESS FILTER

Test Label: IP Address Filtering - AddIPv6AddressFilter

Test Case ID: IPADDRESSFILTERING-5

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Conditional

Profile A Normative Reference: Conditional

Feature Under Test: IP Address Filtering

Test Purpose: To verify that Client is able to add the IP address filter to Device using the AddIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with AddIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes AddIPAddressFilter request message to add the IP address filter on Device.
2. Device responds with code HTTP 200 OK and AddIPAddressFilterResponse message.

Test Result:

NOTE: If Client AddIPAddressFilter request message does not contain "<IPv6Address>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **AddIPAddressFilter** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **AddIPAddressFilter** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<AddIPAddressFilter>" tag after the "<Body>" tag AND
  - [S3] "<AddIPAddressFilter>" includes tag: "<IPv6Address>" AND
  - [S4] "<IPv6Address>" includes tag: "<Address>" with specific IPv6 address value AND
  - [S5] "<IPv6Address>" includes tag: "<PrefixLength>" with value range from "0" to "128" AND
  - [S6] Device response contains "HTTP/* 200 OK" AND

- [S7] Device response contains "<AddIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: IPAddressFiltering.AddIpV6AddressFilter

24.8 REMOVE IPv4 ADDRESS FILTER

Test Label: IP Address Filtering - RemoveIPv4AddressFilter

Test Case ID: IPADDRESSFILTERING-6

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Conditional

Profile A Normative Reference: Conditional

Feature Under Test: IP Address Filtering

Test Purpose: To verify that Client is able to delete the IP address filter from Device using the RemoveIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with RemoveIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes RemoveIPAddressFilter request message to delete the IP address filter from Device.
2. Device responds with code HTTP 200 OK and RemoveIPAddressFilterResponse message.

Test Result:

NOTE: If Client RemoveIPAddressFilter request message does not contain "<IPv4Address>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **RemoveIPAddressFilter** request messages are valid according to XML Schemas listed in Namespaces AND

- Client **RemoveIPAddressFilter** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<RemoveIPAddressFilter>" tag after the "<Body>" tag AND
  - [S3] "<RemoveIPAddressFilter>" includes tag: "<IPv4Address>" AND
  - [S4] "<IPv4Address>" includes tag: "<Address>" with specific IPv4 address value AND
  - [S5] "<IPv4Address>" includes tag: "<PrefixLength>" with value range from "0" to "32" AND
  - [S6] Device response contains "HTTP/* 200 OK" AND
  - [S7] Device response contains "<RemoveIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: IPAddressFiltering.RemoveIpV4AddressFilter

24.9 REMOVE IPv6 ADDRESS FILTER

Test Label: IP Address Filtering - RemoveIPv6AddressFilter

Test Case ID: IPADDRESSFILTERING-7

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Conditional

Profile A Normative Reference: Conditional

Feature Under Test: IP Address Filtering

Test Purpose: To verify that Client is able to delete the IP address filter from Device using the RemoveIPAddressFilter operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with RemoveIPAddressFilter operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes RemoveIPAddressFilter request message to delete the IP address filter from Device.

2. Device responds with code HTTP 200 OK and RemoveIPAddressFilterResponse message.

Test Result:

NOTE: If Client RemoveIPAddressFilter request message does not contain "<IPv6Address>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **RemoveIPAddressFilter** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **RemoveIPAddressFilter** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<RemoveIPAddressFilter>" tag after the "<Body>" tag AND
  - [S3] "<RemoveIPAddressFilter>" includes tag: "<IPv6Address>" AND
  - [S4] "<IPv6Address>" includes tag: "<Address>" with specific IPv6 address value AND
  - [S5] "<IPv6Address>" includes tag: "<PrefixLength>" with value range from "0" to "128" AND
  - [S6] Device response contains "HTTP/* 200 OK" AND
  - [S7] Device response contains "<RemoveIPAddressFilterResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: IPAddressFiltering.RemoveIpV6AddressFilter

25 Persistent Notification Storage Retrieval Test Cases

25.1 Feature Level Requirement:

Validated Feature: PersistentNotificationStorageRetrieval

Check Condition based on Device Features: Persistent Notification Storage is supported by Device.

Required Number of Devices: 1

Profile C Requirement: Conditional

Profile A Requirement: Conditional

25.2 Expected Scenarios Under Test:

1. Client subscribes to device messages using CreatePullPointSubscription operation.
2. Client uses Seek method to change position of the pull pointer to include all NotificationMessages in the persistent storage with UtcTime attribute greater than or equal to the Seek argument.
3. Client uses Pull Point event mechanism to retrieve notification events from Device.
4. Client is considered as supporting Persistent Notification Storage Retrieval if the following conditions are met:
   - Client is able to seek stored events in Device using the Seek operation.
5. Client is considered as NOT supporting Persistent Notification Storage Retrieval if ANY of the following is TRUE:
   - No Valid Device Response to Seek request.

25.3 SEEK

Test Label: Persistent Notification Storage Retrieval - Seek

Test Case ID: PERSISTENTNOTIFICATIONSTORAGERETRIEVAL-1

Profile C Normative Reference: Conditional

Profile A Normative Reference: Conditional

Feature Under Test: Persistent Notification Storage Retrieval

Test Purpose: To verify that Client is able to seek stored events in Device using Pull Point event mechanism and Seek operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with CreatePullPointSubscription, Seek and PullMessages operations present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes CreatePullPointSubscription message.
2. Device responds with code HTTP 200 OK and CreatePullPointSubscriptionResponse message.
3. Client invokes Seek message to re-adjust the pull pointer into the past.
4. Device responds with code HTTP 200 OK and SeekResponse message.
5. Client invokes PullMessages command with Timeout and MessageLimit elements.
6. Device responds with code HTTP 200 OK and PullMessagesResponse message.

Test Result:

PASS -

- Client **CreatePullPointSubscription** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **CreatePullPointSubscription** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<CreatePullPointSubscription>" tag after the "<Body>" tag AND
  - [S2] Device response contains "HTTP/* 200 OK" AND
  - [S3] Device response contains "<CreatePullPointSubscriptionResponse>" tag AND
- Client **Seek** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **Seek** request in Test Procedure fulfills the following requirements:

- [S4] Client request contains "<Seek>" tag after the "<Body>" tag AND
- [S6] Device response contains "HTTP/* 200 OK" AND
- [S7] Device response contains "<SeekResponse>" tag AND

- Client **PullMessages** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **PullMessages** request in Test Procedure fulfills the following requirements:
  - [S8] Client request contains "<PullMessages>" tag after the "<Body>" tag AND
  - [S11] Device response contains "HTTP/* 200 OK" AND
  - [S12] Device response contains "<PullMessagesResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: PersistentNotificationStorageRetrieval.Seek

26 System Date and Time Configuration Test Cases

26.1 Feature Level Requirement:

Validated Feature: SystemDateAndTimeConfiguration

Check Condition based on Device Features: None

Required Number of Devices: 1

Profile A Requirement: Conditional

Profile C Requirement: Optional

Profile G Requirement: Optional

Profile Q Requirement: Conditional

Profile S Requirement: Optional

26.2 Expected Scenarios Under Test:

1. Client connects to Device to configure system date and time.
2. Client is considered as supporting System Date and Time Configuration if the following conditions are met:
   - Client is able to retrieve a system date and time using **GetSystemDateAndTime** operation AND
   - Client is able to configure a system date and time using EITHER **SetSystemDateAndTime** operation OR **SetNTP** operation.
3. Client is considered as NOT supporting System Date and Time Configuration if ANY of the following is TRUE:
   - No valid responses for **GetSystemDateAndTime** request OR
   - No valid responses for **SetSystemDateAndTime** request if detected AND
   - Client does not support NTP feature.

26.3 GET SYSTEM DATE AND TIME

Test Label: System Date and Time Configuration - Get System Date And Time

Test Case ID: SYSTEMDATEANDTIMECONFIGURATION-1

Profile A Normative Reference: Conditional

Profile C Normative Reference: Optional

Profile G Normative Reference: Optional

Profile Q Normative Reference: Conditional

Profile S Normative Reference: Optional

Feature Under Test: Get System Date And Time

Test Purpose: To verify that Device system date and time is received by Client using the **GetSystemDateAndTime** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetSystemDateAndTime** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetSystemDateAndTime** request message to retrieve system date and time from the Device.
2. Device responds with code HTTP 200 OK and **GetSystemDateAndTimeResponse** message.

Test Result:

PASS -

- Client **GetSystemDateAndTime** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetSystemDateAndTime** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:GetSystemDateAndTime** AND
- Device response on the **GetSystemDateAndTime** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:GetSystemDateAndTimeResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: SystemDateAndTimeConfiguration.GetSystemDateAndTime

26.4 SET SYSTEM DATE AND TIME

Test Label: System Date and Time Configuration - Set System Date And Time

Test Case ID: SYSTEMDATEANDTIMECONFIGURATION-2

Profile A Normative Reference: Conditional

Profile C Normative Reference: Optional

Profile G Normative Reference: Optional

Profile Q Normative Reference: Conditional

Profile S Normative Reference: Optional

Feature Under Test: Set System Date And Time

Test Purpose: To verify that Client is able to configure system date and time on Device using the **SetSystemDateAndTime** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetSystemDateAndTime** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetSystemDateAndTime** request message to set Device system date and time.
2. Device responds with code HTTP 200 OK and **SetSystemDateAndTimeResponse** message.

Test Result:

PASS -

- Client **SetSystemDateAndTime** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetSystemDateAndTime** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SetSystemDateAndTime** AND

- [S2] If **tds:DateTimeType** element value is equal to "Manual" THEN **tds:SetSystemDateAndTime** contains **tds:UTCDateTime** element AND

- Device response on the **SetSystemDateAndTime** request fulfills the following requirements:
  - [S3] It has HTTP 200 response code AND
  - [S4] **soapenv:Body** element has child element **tds:SetSystemDateAndTimeResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: SystemDateAndTimeConfiguration.SetSystemDateAndTime

27 HTTP Firmware Upgrade Test Cases

27.1 Feature Level Requirement:

Validated Feature: HTTPFirmwareUpgrade

Check Condition based on Device Features: HTTP Firmware Upgrade is supported by Device.

Required Number of Devices: 1

Profile Q Requirement: Conditional

27.2 Expected Scenarios Under Test:

1. Client connects to the Device to instruct it to prepare for upgrade using the StartFirmwareUpgrade operation.
2. Client sends the firmware image using HTTP POST to the upload URI provided by the Device in StartFirmwareUpgradeResponse.
3. Client is considered as supporting HTTP Firmware Upgrade if the following conditions are met:
   - Client is able to instruct the Device to prepare for upgrade using **StartFirmwareUpgrade** operation if Device supports HTTP Firmware Upgrade AND
   - Client is able to send the firmware image using **HTTP POST** if Device supports HTTP Firmware Upgrade.
4. Client is considered as NOT supporting HTTP Firmware Upgrade if ANY of the following is TRUE:
   - No valid responses for **StartFirmwareUpgrade** request if Device supports HTTP Firmware Upgrade OR
   - No valid **HTTP POST** request to the upload URI if Device supports HTTP Firmware Upgrade.
   - No valid responses for **HTTP POST** request to the upload URI with firmware image if Device supports HTTP Firmware Upgrade.

27.3 FIRMWARE UPGRADE VIA HTTP

Test Label: Firmware Upgrade via HTTP - Start Firmware Upgrade

Test Case ID: HTTPFIRMWAREUPGRADE-1

Profile Q Normative Reference: Conditional

Feature Under Test: Start Firmware Upgrade

Test Purpose: To verify that Client is able to upgrade the Device firmware via HTTP using the **StartFirmwareUpgrade** operation and **HTTP POST**.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **StartFirmwareUpgrade** operation present.
- Device supports Http Firmware Upgrade.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **StartFirmwareUpgrade** request message to instruct the Device to prepare for upgrade.
2. Device responds with code HTTP 200 OK and **StartFirmwareUpgradeResponse** message.
3. Client sends the firmware image using **HTTP POST** to the upload URI provided by the Device in StartFirmwareUpgradeResponse.
4. Device responds with code HTTP 200 OK message.

Test Result:

PASS -

- Client **StartFirmwareUpgrade** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **StartFirmwareUpgrade** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:StartFirmwareUpgrade** AND
- Device response on the **StartFirmwareUpgrade** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:StartFirmwareUpgradeResponse**.
- There is **HTTP POST** request in Test Procedure fulfills the following requirements:

- [S4] It invoked to address which equal to **tds:StartFirmwareUpgradeResponse/ tds:UploadUri** value from the Device response to **StartFirmwareUpgrade** request AND
- [S5] It invoked after the Client **StartFirmwareUpgrade** request AND
- [S6] It contains HTTP Content-Type Header with value is equal to “application/octet-stream” AND

- Device response on the **HTTP POST** request fulfills the following requirements:
  - [S7] It has HTTP 200 response code.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: HTTPFirmwareUpgrade.HTTPFirmwareUpgrade

28 HTTP System Backup Test Cases

28.1 Feature Level Requirement:

Validated Feature: HTTPSystemBackup

Check Condition based on Device Features: HTTP System Backup is supported by Device.

Required Number of Devices: 1

Profile Q Requirement: Conditional

28.2 Expected Scenarios Under Test:

1. Client connects to the Device to retrieve URI from which a system backup may be downloaded using the GetSystemUris operation.

   Client gets the backup system configurations using HTTP GET sent to the System Backup Uri provided by the Device in GetSystemUrisResponse.

2. Client is considered as supporting HTTP System Backup if the following conditions are met:

   - Client is able to retrieve URI from Device for system backup using **GetSystemUris** operation if Device supports HTTP System Backup AND
   - Client is able to to backup system configurations using **HTTP GET** if Device supports HTTP System Backup AND

3. Client is considered as NOT supporting HTTP System Backup if ANY of the following is TRUE:

   - No valid responses for **GetSystemUris** request if Device supports HTTP System Backup OR
   - No valid responses for **HTTP GET** request to the System Backup Uri if Device supports HTTP System Backup.

28.3 HTTP SYSTEM BACKUP

Test Label: System Backup via HTTP - Get System Uris

Test Case ID: HTTPSYSTEMBACKUP-1

Profile Q Normative Reference: Conditional

Feature Under Test: Get System Uris

Test Purpose: To verify that Client is able to backup system configurations via HTTP using the **GetSystemUris** operation and **HTTP GET**.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetSystemUris** operation present.
- Device supports HTTP System Backup.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetSystemUris** request message to retrieve URI from which a system backup file may be downloaded.
2. Device responds with code HTTP 200 OK and **GetSystemUrisResponse** message.
3. Client retrieves the backup file using **HTTP GET** to the System Backup Uri provided by the Device in GetSystemUrisResponse.
4. Device responds with code HTTP 200 OK message and with backup file.

Test Result:

PASS -

- Client **GetSystemUris** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetSystemUris** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:GetSystemUris** AND
- Device response on the **GetSystemUris** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:GetSystemUrisResponse**.
- There is **HTTP GET** request in Test Procedure that fulfills the following requirements:
  - [S4] It invoked to address which equal to **tds:GetSystemUrisResponse/ tds:SystemBackupUri** value from the Device response to **GetSystemUris** request AND
  - [S5] It invoked after the Client **GetSystemUris** request AND
- Device response on the **HTTP GET** request fulfills the following requirements:

- [S6] It has HTTP 200 response code.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: HTTPSystemBackup.HTTPSystemBackup

29 HTTP System Restore Test Cases

29.1 Feature Level Requirement:

Validated Feature: HTTPSystemRestore

Check Condition based on Device Features: HTTP System Backup is supported by Device.

Required Number of Devices: 1

Profile Q Requirement: Conditional

29.2 Expected Scenarios Under Test:

1. Client connects to the Device to retrieve URI to which the backuped data may be uploaded using the StartSystemRestore operation.

   Client uploads the backuped configuration data using HTTP POST to the Upload Uri provided by the Device in StartSystemRestoreResponse.

2. Client is considered as supporting HTTP System Restore if the following conditions are met:

   - Client is able to retrieve URI from Device for restore system configurations using **StartSystemRestore** operation if Device supports HTTP System Backup AND
   - Client is able to send the backuped data to the Device using **HTTP POST** if Device supports HTTP System Backup.

3. Client is considered as NOT supporting HTTP System Restore if ANY of the following is TRUE:

   - No valid responses for **StartSystemRestore** request if Device supports HTTP System Backup OR
   - No valid **HTTP POST** request to the Upload Uri if Device supports HTTP System Backup.
   - No valid responses for **HTTP POST** request to the Upload Uri if Device supports HTTP System Backup.

29.3 HTTP SYSTEM RESTORE

Test Label: System Restore via HTTP - Start System Restore

Test Case ID: HTTPSYSTEMRESTORE-1

Profile Q Normative Reference: Conditional

Feature Under Test: Start System Restore

Test Purpose: To verify that Client is able to restore system configurations via HTTP using the **StartSystemRestore** operation and **HTTP POST**.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **StartSystemRestore** operation present.
- Device supports HTTP System Backup.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **StartSystemRestore** request message to retrieve upload URI from the Device.
2. Device responds with code HTTP 200 OK and **StartSystemRestoreResponse** message.
3. Client transmits the configuration data to the upload URI using **HTTP POST**.
4. Device responds with code HTTP 200 OK message.

Test Result:

PASS -

- Client **StartSystemRestore** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **StartSystemRestore** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:StartSystemRestore** AND
- Device response on the **StartSystemRestore** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:StartSystemRestoreResponse**.
- There is **HTTP POST** request in Test Procedure that fulfills the following requirements:
  - [S4] It invoked to address which equal to **tds:StartSystemRestore/tds:UploadUri** value from the Device response to **StartSystemRestore** request AND
  - [S5] It invoked after the Client **StartSystemRestore** request AND

- [S6] It contains HTTP Content-Type Header with value is equal to “application/octet-stream” AND

- Device response on the **HTTP POST** request fulfills the following requirements:
  - [S7] It has HTTP 200 response code.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: HTTPSystemRestore.HTTPSystemRestore

30 Monitoring Notifications Test Cases

30.1 Feature Level Requirement:

Validated Feature: MonitoringNotifications

Check Condition based on Device Features: Monitoring/ProcessorUsage or Monitoring/OperatingTime/LastReset or Monitoring/OperatingTime/LastReboot or Monitoring/OperatingTime/LastClockSynchronization is supported by Device.

Required Number of Devices: 1

Profile Q Requirement: Conditional

30.2 Expected Scenarios Under Test:

1. Client subscribes to device messages using **CreatePullPointSubscription** operation to get monitoring notifications.
2. Client uses Pull Point event mechanism to retrieve notification events from Device.
3. Client is considered as supporting Monitoring Notifications if the following conditions are met:
   - Client supports EventHandling_Pullpoint feature AND
   - Client is able to retrieve at least one of the following notifications:
     - tns1:Monitoring/ProcessorUsage notification about processor usage if Device supports MonitoringProcessorUsageEvent feature
     - tns1:Monitoring/OperatingTime/LastReset notification about last reset if Device supports MonitoringOperatingTimeLastResetEvent feature
     - tns1:Monitoring/OperatingTime/LastReboot notification about last reboot if Device supports MonitoringOperatingTimeLastRebootEvent feature
     - tns1:Monitoring/OperatingTime/LastClockSynchronization notification about last clock synchronization if Device supports MonitoringOperatingTimeLastClockSynchronizationEvent feature
4. Client is considered as NOT supporting Monitoring Notifications if ANY of the following is TRUE:
   - Client does not support EventHandling_Pullpoint feature OR

- Client is not able to retrieve the following notifications:
  - tns1:Monitoring/ProcessorUsage notification about processor usage if Device supports MonitoringProcessorUsageEvent feature
  - tns1:Monitoring/OperatingTime/LastReset notification about last reset if Device supports MonitoringOperatingTimeLastResetEvent feature
  - tns1:Monitoring/OperatingTime/LastReboot notification about last reboot if Device supports MonitoringOperatingTimeLastRebootEvent feature
  - tns1:Monitoring/OperatingTime/LastClockSynchronization notifications about last clock synchronization if Device supports MonitoringOperatingTimeLastClockSynchronizationEvent feature.

31 Device Management Notifications Test Cases

31.1 Feature Level Requirement:

Validated Feature: DeviceManagementNotifications

Check Condition based on Device Features: Device/HardwareFailure/FanFailure or Device/HardwareFailure/PowerSupplyFailure or Device/HardwareFailure/StorageFailure or Device/HardwareFailure/TemperatureCritical or Monitoring/Backup/Last is supported by Device.

Required Number of Devices: 1

Profile Q Requirement: Conditional

31.2 Expected Scenarios Under Test:

1. Client subscribes to device messages using **CreatePullPointSubscription** operation to get device management notifications.
2. Client uses Pull Point event mechanism to retrieve notification events from Device.
3. Client is considered as supporting Device Management Notifications if the following conditions are met:
   - Client supports EventHandling_Pullpoint feature AND
   - Client is able to retrieve at least one of the following notifications:
     - tns1:Device/HardwareFailure/FanFailure notification about fan failure if Device supports DeviceHardwareFailureFanFailureEvent feature
     - tns1:Device/HardwareFailure/PowerSupplyFailure notification about power supply failure if Device supports DeviceHardwareFailurePowerSupplyFailureEvent feature
     - tns1:Device/HardwareFailure/StorageFailure notification about storage failure if Device supports DeviceHardwareFailureStorageFailureEvent feature
     - tns1:Device/HardwareFailure/TemperatureCritical notification about temperature critical if Device supports DeviceHardwareFailureTemperatureCriticalEvent feature
     - tns1:Monitoring/Backup/Last notification about last backup if Device supports MonitoringBackupLastEvent feature
4. Client is considered as NOT supporting Device Management Notifications if ANY of the following is TRUE:

- Client does not support EventHandling_Pullpoint feature OR
- Client is not able to retrieve the following notifications:
  - tns1:Device/HardwareFailure/FanFailure notification about fan failure if Device supports DeviceHardwareFailureFanFailureEvent feature
  - tns1:Device/HardwareFailure/PowerSupplyFailure notification about power supply failure if Device supports DeviceHardwareFailurePowerSupplyFailureEvent feature
  - tns1:Device/HardwareFailure/StorageFailure notification about storage failure if Device supports DeviceHardwareFailureStorageFailureEvent feature
  - tns1:Device/HardwareFailure/TemperatureCritical notification about temperature critical if Device supports DeviceHardwareFailureTemperatureCriticalEvent feature
  - tns1:Monitoring/Backup/Last notification about last backup if Device supports MonitoringBackupLastEvent feature

32 Hostname Configuration Test Cases

32.1 Feature Level Requirement:

Validated Feature: HostnameConfiguration

Check Condition based on Device Features: None

Required Number of Devices: 1

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile G Requirement: Optional

Profile Q Requirement: Conditional

Profile S Requirement: Optional

32.2 Expected Scenarios Under Test:

1. Client connects to Device to configure hostname.
2. Client is considered as supporting Hostname Configuration if the following conditions are met:
   - Client is able to retrieve a hostname information from the Device using **GetHostname** operation AND
   - Client is able set a network hostname on the Device using **SetHostname** operation.
3. Client is considered as NOT supporting Hostname Configuration if ANY of the following is TRUE:
   - No valid responses for **GetHostname** request OR
   - No valid responses for **SetHostname** request.

32.3 GET HOSTNAME

Test Label: Hostname Configuration - Get Hostname

Test Case ID: HOSTNAMECONFIGURATION-1

Profile A Normative Reference: Optional

Profile C Normative Reference: Optional

Profile G Normative Reference: Optional

Profile Q Normative Reference: Conditional

Profile S Normative Reference: Optional

Feature Under Test: Get Hostname

Test Purpose: To verify that hostname settings of the Device are received by Client using the **GetHostname** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetHostname** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetHostname** request message to retrieve hostname from the Device.
2. Device responds with code HTTP 200 OK and **GetHostnameResponse** message.

Test Result:

PASS -

- Client **GetHostname** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetHostname** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:GetHostname** AND
- Device response on the **GetHostname** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:GetHostnameResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: HostnameConfiguration.GetHostname

32.4 SET HOSTNAME

Test Label: Hostname Configuration - Set Hostname

Test Case ID: HOSTNAMECONFIGURATION-2

Profile A Normative Reference: Optional

Profile C Normative Reference: Optional

Profile G Normative Reference: Optional

Profile Q Normative Reference: Conditional

Profile S Normative Reference: Optional

Feature Under Test: Set Hostname

Test Purpose: To verify that Client is able to set the Hostname settings on Device using the **SetHostname** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetHostname** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetHostname** request message to set hostname on the Device.
2. Device responds with code HTTP 200 OK and **SetHostnameResponse** message.

Test Result:

PASS -

- Client **SetHostname** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetHostname** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SetHostname** AND
- Device response on the **SetHostname** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:SetHostnameResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: HostnameConfiguration.SetHostname

33 DNS Configuration Test Cases

33.1 Feature Level Requirement:

Validated Feature: DNSConfiguration

Check Condition based on Device Features: None

Required Number of Devices: 1

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile G Requirement: Optional

Profile Q Requirement: Conditional

Profile S Requirement: Optional

33.2 Expected Scenarios Under Test:

1. Client connects to Device to configure a domain name server.
2. Client is considered as supporting DNS Configuration if the following conditions are met:
   - Client is able to get DNS settings from the Device using **GetDNS** operation AND
   - Client is able set DNS settings on the Device using **SetDNS** operation.
3. Client is considered as NOT supporting DNS Configuration if ANY of the following is TRUE:
   - No valid responses for **GetDNS** request OR
   - No valid responses for **SetDNS** request.

33.3 GET DNS

Test Label: DNS Configuration - Get DNS

Test Case ID: DNSCONFIGURATION-1

Profile A Normative Reference: Optional

Profile C Normative Reference: Optional

Profile G Normative Reference: Optional

Profile Q Normative Reference: Conditional

Profile S Normative Reference: Optional

Feature Under Test: Get DNS

Test Purpose: To verify that DNS settings of Device are received by Client using the **GetDNS** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetDNS** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetDNS** request message to retrieve DNS settings from the Device.
2. Device responds with code HTTP 200 OK and **GetDNSResponse** message.

Test Result:

PASS -

- Client **GetDNS** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetDNS** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:GetDNS** AND
- Device response on the **GetDNS** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:GetDNSResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: DNSConfiguration.GetDNS

33.4 SET DNS

Test Label: DNS Configuration - Set DNS

Test Case ID: DNSCONFIGURATION-2

Profile A Normative Reference: Optional

Profile C Normative Reference: Optional

Profile G Normative Reference: Optional

Profile Q Normative Reference: Conditional

Profile S Normative Reference: Optional

Feature Under Test: Set DNS

Test Purpose: To verify that Client is able to set the DNS settings on Device using the **SetDNS** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetDNS** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetDNS** request message to set hostname on the Device.
2. Device responds with code HTTP 200 OK and **SetDNSResponse** message.

Test Result:

PASS -

- Client **SetDNS** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetDNS** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SetDNS** AND
- Device response on the **SetDNS** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:SetDNSResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: DNSConfiguration.SetDNS

34 Network Protocols Configuration Test Cases

34.1 Feature Level Requirement:

Validated Feature: NetworkProtocolsConfiguration

Check Condition based on Device Features: None

Required Number of Devices: 1

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile G Requirement: Optional

Profile Q Requirement: Conditional

Profile S Requirement: Optional

34.2 Expected Scenarios Under Test:

1. Client connects to Device to configure a network protocols.
2. Client is considered as supporting Network Protocols Configuration if the following conditions are met:
   - Client is able to get defined network protocols from the Device using **GetNetworkProtocols** operation AND
   - Client is able configures defined network protocols on the Device using **SetNetworkProtocols** operation.
3. Client is considered as NOT supporting Network Protocols Configuration if ANY of the following is TRUE:
   - No valid responses for **GetNetworkProtocols** request OR
   - No valid responses for **SetNetworkProtocols** request.

34.3 GET NETWORK PROTOCOLS

Test Label: Network Protocols Configuration - Get Network Protocols

Test Case ID: NETWORKPROTOCOLSCONFIGURATION-1

Profile A Normative Reference: Optional

Profile C Normative Reference: Optional

Profile G Normative Reference: Optional

Profile Q Normative Reference: Conditional

Profile S Normative Reference: Optional

Feature Under Test: Get Network Protocols

Test Purpose: To verify that network protocols of Device are received by Client using the **GetNetworkProtocols** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **GetNetworkProtocols** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **GetNetworkProtocols** request message to retrieve network protocols from the Device.
2. Device responds with code HTTP 200 OK and **GetNetworkProtocolsResponse** message.

Test Result:

PASS -

- Client **GetNetworkProtocols** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **GetNetworkProtocols** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:GetNetworkProtocols** AND
- Device response on the **GetNetworkProtocols** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:GetNetworkProtocolsResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: NetworkProtocolsConfiguration.GetNetworkProtocols

34.4 SET NETWORK PROTOCOLS

Test Label: Network Protocols Configuration - Set Network Protocols

Test Case ID: NETWORKPROTOCOLSCONFIGURATION-2

Profile A Normative Reference: Optional

Profile C Normative Reference: Optional

Profile G Normative Reference: Optional

Profile Q Normative Reference: Conditional

Profile S Normative Reference: Optional

Feature Under Test: Set Network Protocols

Test Purpose: To verify that Client is able to configure defined network protocols on Device using the **SetNetworkProtocols** operation.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with **SetNetworkProtocols** operation present.

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SetNetworkProtocols** request message to set hostname on the Device.
2. Device responds with code HTTP 200 OK and **SetNetworkProtocolsResponse** message.

Test Result:

PASS -

- Client **SetNetworkProtocols** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetNetworkProtocols** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SetNetworkProtocols** AND
- Device response on the **SetNetworkProtocols** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:SetNetworkProtocolsResponse**.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: NetworkProtocolsConfiguration.SetNetworkProtocols

35 HTTP Digest Authentication for RTSP Test Cases

35.1 Feature Level Requirement:

Validated Feature: HTTP Digest Authentication for RTSP

Check Condition based on Device Features: Profile T

Required Number of Devices: TO BE DISCUSSED

Profile S Requirement: None

Profile G Requirement: None

Profile A Requirement: None

Profile C Requirement: None

Profile Q Requirement: None

Profile T Requirement: Mandatory

35.2 Expected Scenarios Under Test:

1. Client invokes a specific RTSP command which is under testing without any user credentials (no UsernameToken, no HTTP Digest authentication header).
2. IF Device returns HTTP 401 Unauthorized error along with WWW-Authentication: Digest header, then Client resends RTSP command with WWW-Authenticate header.
3. Client is considered as supporting HTTP Digest Authentication for RTSP if the following conditions are met:
   - Device returns a valid response to specific RTSP request with HTTP Digest authentication header.
4. Client is considered as NOT supporting HTTP Digest Authentication for RTSP if the following is TRUE:
   - All HTTP Digest attempts detected for RTSP are failed.

35.3 HTTP DIGEST AUTHENTICATION FOR RTSP

Test Label: HTTP Digest For RTSP

Test Case ID: HTTPDIGESTFORRTSP-1

Profile A Normative Reference: None

Profile C Normative Reference: None

Profile G Normative Reference: None

Profile Q Normative Reference: None

Profile T Normative Reference: Mandatory

Feature Under Test: HTTP Digest

Test Purpose: To verify that the Client supports the HTTP Digest Authentication for RTSP level security.

Pre-Requisite:

- The Network Trace Capture files contains at least one Conversation between Client and Device with HTTP Digest Authentication for RTSP commands present

Test Procedure (expected to be reflected in network trace file):

1. Client sends a RTSP request that requires authentication (e.g. DESCRIBE) to the Device without any authentication.
2. Device rejects the request with a RTSP 401 status code, AND a WWW-Authenticate Response Header.
3. Client re-sends the RTSP request with a Authorization Request Header.
4. Device accepts the correct request with RTSP 200 OK status code.

Test Result:

PASS -

- There is Client RTSP request in Test Procedure that does not contain any authentication AND
- Device response on the Client RTSP request fulfills the following requirements:
  - It has RTSP 401 status code AND
  - WWW-Authenticate Response Header contains challenge = "Digest" element AND
  - WW-Authenticate Response Header contains "realm=*" element AND
  - WW-Authenticate Response Header contains "nonce=*" element AND

- There is Client RTSP request in Test Procedure that fulfills the following requirements
  - WW-Authenticate Request Header credentials = "Digest" element AND
  - WW-Authenticate Request Header contains "realm=*" element with value from Device response AND
  - WW-Authenticate Request Header contains "nonce=*" element with value from Device response AND
  - WW-Authenticate Request Header contains "uri=*" element AND
- Device responds with code RTSP 200 OK.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: HTTP_Digest_for_RTSP.HTTP_Digest_for_RTSP

36 Auxiliary Commands Test Cases

36.1 Feature Level Requirement:

Validated Feature: AuxiliaryCommands

Check Condition based on Device Features: None (ONVIF Profile T Simulator is used as device).

Required Number of Devices: 1

Profile S Requirement: None

Profile G Requirement: None

Profile A Requirement: None

Profile C Requirement: None

Profile Q Requirement: None

Profile T Requirement: Conditional

36.2 Expected Scenarios Under Test:

Recommendation: Clients should support all auxiliary operations listed in the scenario.

1. Client connects to ONVIF Profile T Simulator to manage auxiliary commands supported by the Device.
2. Client is considered as supporting Auxiliary Commands if the following conditions are met:
   - ONVIF Profile T Simulator detects **SendAuxiliaryCommand** request with at least one of the following auxiliary commands:
     - tt:Wiper|On
     - tt:Wiper|Off
     - tt:Washer|On
     - tt:Washer|Off
     - tt:WashingProcedure|On
     - tt:WashingProcedure|Off

- tt:IRLamp|On
- tt:IRLamp|Off
- tt:IRLamp|Auto

3. Client is considered as NOT supporting Auxiliary Commands if ANY of the following is TRUE:

- <listitem>

  ONVIF Profile T Simulator does not detect **SendAuxiliaryCommand** request with tt:Wiper|On auxiliary command AND
  </listitem>
  <listitem>

  ONVIF Profile T Simulator does not detect **SendAuxiliaryCommand** request with tt:Wiper|Off auxiliary command AND
  </listitem>
  <listitem>

  ONVIF Profile T Simulator does not detect **SendAuxiliaryCommand** request with tt:Washer|On auxiliary command AND
  </listitem>
  <listitem>

  ONVIF Profile T Simulator does not detect **SendAuxiliaryCommand** request with tt:Washer|Off auxiliary command AND
  </listitem>
  <listitem>

  ONVIF Profile T Simulator does not detect **SendAuxiliaryCommand** request with tt:WashingProcedure|On auxiliary command AND
  </listitem>
  <listitem>

  ONVIF Profile T Simulator does not detect **SendAuxiliaryCommand** request with tt:WashingProcedure|Off auxiliary command AND
  </listitem>
  <listitem>

  ONVIF Profile T Simulator does not detect **SendAuxiliaryCommand** request with tt:IRLamp|On auxiliary command AND
  </listitem>

<listitem>

ONVIF Profile T Simulator does not detect **SendAuxiliaryCommand** request with tt:IRLamp|Off auxiliary command AND
</listitem>
<listitem>

ONVIF Profile T Simulator does not detect **SendAuxiliaryCommand** request with tt:IRLamp|Auto auxiliary command.
</listitem>

36.3 WIPER ON

Test Label: Auxiliary Commands

Test Case ID: AUXILIARYCOMMANDS-1

Profile A Normative Reference: None

Profile C Normative Reference: None

Profile G Normative Reference: None

Profile Q Normative Reference: None

Profile T Normative Reference: Conditional

Feature Under Test: Send Auxiliary Commands

Test Purpose: To verify that the Client supports the SendAuxiliaryCommand operation.

Pre-Requisite:

- The .osc file of conversation with ONVIF Profile T Simulator is present (please, refer to 'Profile T Simulator' section in Help) .

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SendAuxiliaryCommand** request message with **AuxiliaryCommand** = **"tt:Wiper|On"** to the ONVIF Profile T Simulator (please, refer to 'Profile T Simulator' section in Help).
2. ONVIF Profile T Simulator responds with **SendAuxiliaryCommandResponse** message and 200 OK HTTP code.

Test Result:

PASS -

- Client **SendAuxiliaryCommand** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SendAuxiliaryCommand** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SendAuxiliaryCommand** AND
  - [S2] **tds:SendAuxiliaryCommand/tds:AuxiliaryCommand** is equal to **"tt:Wiper|On"** AND
- ONVIF Profile T Simulator response on the **SendAuxiliaryCommand** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:SendAuxiliaryCommandResponse**.

FAIL -

- None.

Validated Feature List: AuxiliaryCommands_WiperOn

36.4 WIPER OFF

Test Label: Auxiliary Commands

Test Case ID: AUXILIARYCOMMANDS-2

Profile A Normative Reference: None

Profile C Normative Reference: None

Profile G Normative Reference: None

Profile Q Normative Reference: None

Profile T Normative Reference: Conditional

Feature Under Test: Send Auxiliary Commands

Test Purpose: To verify that the Client supports the SendAuxiliaryCommand operation.

Pre-Requisite:

- The .osc file of conversation with ONVIF Profile T Simulator is present (please, refer to 'Profile T Simulator' section in Help) .

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SendAuxiliaryCommand** request message with **AuxiliaryCommand** = **"tt:Wiper|Off"** to the ONVIF Profile T Simulator (please, refer to 'Profile T Simulator' section in Help).
2. ONVIF Profile T Simulator responds with **SendAuxiliaryCommandResponse** message and 200 OK HTTP code.

Test Result:

PASS -

- Client **SendAuxiliaryCommand** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SendAuxiliaryCommand** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SendAuxiliaryCommand** AND
  - [S2] **tds:SendAuxiliaryCommand/tds:AuxiliaryCommand** is equal to **"tt:Wiper|Off"** AND
- ONVIF Profile T Simulator response on the **SendAuxiliaryCommand** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:SendAuxiliaryCommandResponse**.

FAIL -

- None.

Validated Feature List: AuxiliaryCommands_WiperOff

36.5 WASHER ON

Test Label: Auxiliary Commands

Test Case ID: AUXILIARYCOMMANDS-3

Profile A Normative Reference: None

Profile C Normative Reference: None

Profile G Normative Reference: None

Profile Q Normative Reference: None

Profile T Normative Reference: Conditional

Feature Under Test: Send Auxiliary Commands

Test Purpose: To verify that the Client supports the SendAuxiliaryCommand operation.

Pre-Requisite:

- The .osc file of conversation with ONVIF Profile T Simulator is present (please, refer to 'Profile T Simulator' section in Help) .

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SendAuxiliaryCommand** request message with **AuxiliaryCommand** = **"tt:Washer|On"** to the ONVIF Profile T Simulator (please, refer to 'Profile T Simulator' section in Help).
2. ONVIF Profile T Simulator responds with **SendAuxiliaryCommandResponse** message and 200 OK HTTP code.

Test Result:

PASS -

- Client **SendAuxiliaryCommand** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SendAuxiliaryCommand** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SendAuxiliaryCommand** AND
  - [S2] **tds:SendAuxiliaryCommand/tds:AuxiliaryCommand** is equal to **"tt:Washer|On"** AND
- ONVIF Profile T Simulator response on the **SendAuxiliaryCommand** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:SendAuxiliaryCommandResponse**.

FAIL -

- None.

Validated Feature List: AuxiliaryCommands_WasherOn

36.6 WASHER OFF

Test Label: Auxiliary Commands

Test Case ID: AUXILIARYCOMMANDS-4

Profile A Normative Reference: None

Profile C Normative Reference: None

Profile G Normative Reference: None

Profile Q Normative Reference: None

Profile T Normative Reference: Conditional

Feature Under Test: Send Auxiliary Commands

Test Purpose: To verify that the Client supports the SendAuxiliaryCommand operation.

Pre-Requisite:

- The .osc file of conversation with ONVIF Profile T Simulator is present (please, refer to 'Profile T Simulator' section in Help) .

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SendAuxiliaryCommand** request message with **AuxiliaryCommand** = **"tt:Washer|Off"** to the ONVIF Profile T Simulator (please, refer to 'Profile T Simulator' section in Help).
2. ONVIF Profile T Simulator responds with **SendAuxiliaryCommandResponse** message and 200 OK HTTP code.

Test Result:

PASS -

- Client **SendAuxiliaryCommand** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SendAuxiliaryCommand** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SendAuxiliaryCommand** AND
  - [S2] **tds:SendAuxiliaryCommand/tds:AuxiliaryCommand** is equal to **"tt:Washer|Off"** AND

- ONVIF Profile T Simulator response on the **SendAuxiliaryCommand** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:SendAuxiliaryCommandResponse**.

FAIL -

- None.

Validated Feature List: AuxiliaryCommands_WasherOff

36.7 WASHINGPROCEDURE ON

Test Label: Auxiliary Commands

Test Case ID: AUXILIARYCOMMANDS-5

Profile A Normative Reference: None

Profile C Normative Reference: None

Profile G Normative Reference: None

Profile Q Normative Reference: None

Profile T Normative Reference: Conditional

Feature Under Test: Send Auxiliary Commands

Test Purpose: To verify that the Client supports the SendAuxiliaryCommand operation.

Pre-Requisite:

- The .osc file of conversation with ONVIF Profile T Simulator is present (please, refer to 'Profile T Simulator' section in Help) .

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SendAuxiliaryCommand** request message with **AuxiliaryCommand** = **"tt:WashingProcedure|On"** to the ONVIF Profile T Simulator (please, refer to 'Profile T Simulator' section in Help).
2. ONVIF Profile T Simulator responds with **SendAuxiliaryCommandResponse** message and 200 OK HTTP code.

Test Result:

PASS -

- Client **SendAuxiliaryCommand** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SendAuxiliaryCommand** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SendAuxiliaryCommand** AND
  - [S2] **tds:SendAuxiliaryCommand/tds:AuxiliaryCommand** is equal to **"tt:WashingProcedure|On"** AND
- ONVIF Profile T Simulator response on the **SendAuxiliaryCommand** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:SendAuxiliaryCommandResponse**.

FAIL -

- None.

Validated Feature List: AuxiliaryCommands_WashingProcedureOn

36.8 WASHINGPROCEDURE OFF

Test Label: Auxiliary Commands

Test Case ID: AUXILIARYCOMMANDS-6

Profile A Normative Reference: None

Profile C Normative Reference: None

Profile G Normative Reference: None

Profile Q Normative Reference: None

Profile T Normative Reference: Conditional

Feature Under Test: Send Auxiliary Commands

Test Purpose: To verify that the Client supports the SendAuxiliaryCommand operation.

Pre-Requisite:

- The .osc file of conversation with ONVIF Profile T Simulator is present (please, refer to 'Profile T Simulator' section in Help) .

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SendAuxiliaryCommand** request message with **AuxiliaryCommand** = **"tt:WashingProcedure|Off"** to the ONVIF Profile T Simulator (please, refer to 'Profile T Simulator' section in Help).
2. ONVIF Profile T Simulator responds with **SendAuxiliaryCommandResponse** message and 200 OK HTTP code.

Test Result:

PASS -

- Client **SendAuxiliaryCommand** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SendAuxiliaryCommand** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SendAuxiliaryCommand** AND
  - [S2] **tds:SendAuxiliaryCommand/tds:AuxiliaryCommand** is equal to **"tt:WashingProcedure|Off"** AND
- ONVIF Profile T Simulator response on the **SendAuxiliaryCommand** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:SendAuxiliaryCommandResponse**.

FAIL -

- None.

Validated Feature List: AuxiliaryCommands_WashingProcedureOff

36.9 IRLAMP ON

Test Label: Auxiliary Commands

Test Case ID: AUXILIARYCOMMANDS-7

Profile A Normative Reference: None

Profile C Normative Reference: None

Profile G Normative Reference: None

Profile Q Normative Reference: None

Profile T Normative Reference: Conditional

Feature Under Test: Send Auxiliary Commands

Test Purpose: To verify that the Client supports the SendAuxiliaryCommand operation.

Pre-Requisite:

- The .osc file of conversation with ONVIF Profile T Simulator is present (please, refer to 'Profile T Simulator' section in Help) .

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SendAuxiliaryCommand** request message with **AuxiliaryCommand** = **"tt:IRLamp|On"** to the ONVIF Profile T Simulator (please, refer to 'Profile T Simulator' section in Help).
2. ONVIF Profile T Simulator responds with **SendAuxiliaryCommandResponse** message and 200 OK HTTP code.

Test Result:

PASS -

- Client **SendAuxiliaryCommand** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SendAuxiliaryCommand** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SendAuxiliaryCommand** AND
  - [S2] **tds:SendAuxiliaryCommand/tds:AuxiliaryCommand** is equal to **"tt:IRLamp|On"** AND
- ONVIF Profile T Simulator response on the **SendAuxiliaryCommand** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:SendAuxiliaryCommandResponse**.

FAIL -

- None.

Validated Feature List: AuxiliaryCommands_IRLampOn

36.10 IRLAMP OFF

Test Label: Auxiliary Commands

Test Case ID: AUXILIARYCOMMANDS-8

Profile A Normative Reference: None

Profile C Normative Reference: None

Profile G Normative Reference: None

Profile Q Normative Reference: None

Profile T Normative Reference: Conditional

Feature Under Test: Send Auxiliary Commands

Test Purpose: To verify that the Client supports the SendAuxiliaryCommand operation.

Pre-Requisite:

- The .osc file of conversation with ONVIF Profile T Simulator is present (please, refer to 'Profile T Simulator' section in Help) .

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SendAuxiliaryCommand** request message with **AuxiliaryCommand** = **"tt:IRLamp|Off"** to the ONVIF Profile T Simulator (please, refer to 'Profile T Simulator' section in Help).
2. ONVIF Profile T Simulator responds with **SendAuxiliaryCommandResponse** message and 200 OK HTTP code.

Test Result:

PASS -

- Client **SendAuxiliaryCommand** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SendAuxiliaryCommand** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SendAuxiliaryCommand** AND

- [S2] **tds:SendAuxiliaryCommand/tds:AuxiliaryCommand** is equal to **"tt:IRLamp|Off"** AND

- ONVIF Profile T Simulator response on the **SendAuxiliaryCommand** request fulfills the following requirements:

  - [S2] It has HTTP 200 response code AND

  - [S3] **soapenv:Body** element has child element **tds:SendAuxiliaryCommandResponse**.

FAIL -

- None.

Validated Feature List: AuxiliaryCommands_IRLampOff

36.11 IRLAMP AUTO

Test Label: Auxiliary Commands

Test Case ID: AUXILIARYCOMMANDS-9

Profile A Normative Reference: None

Profile C Normative Reference: None

Profile G Normative Reference: None

Profile Q Normative Reference: None

Profile T Normative Reference: Conditional

Feature Under Test: Send Auxiliary Commands

Test Purpose: To verify that the Client supports the SendAuxiliaryCommand operation.

Pre-Requisite:

- The .osc file of conversation with ONVIF Profile T Simulator is present (please, refer to 'Profile T Simulator' section in Help) .

Test Procedure (expected to be reflected in network trace file):

1. Client invokes **SendAuxiliaryCommand** request message with **AuxiliaryCommand** = **"tt:IRLamp|Auto"** to the ONVIF Profile T Simulator (please, refer to 'Profile T Simulator' section in Help).

2. ONVIF Profile T Simulator responds with **SendAuxiliaryCommandResponse** message and 200 OK HTTP code.

Test Result:

PASS -

- Client **SendAuxiliaryCommand** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SendAuxiliaryCommand** request in Test Procedure fulfills the following requirements:
  - [S1] **soapenv:Body** element has child element **tds:SendAuxiliaryCommand** AND
  - [S2] **tds:SendAuxiliaryCommand/tds:AuxiliaryCommand** is equal to **"tt:IRLamp|Auto"** AND
- ONVIF Profile T Simulator response on the **SendAuxiliaryCommand** request fulfills the following requirements:
  - [S2] It has HTTP 200 response code AND
  - [S3] **soapenv:Body** element has child element **tds:SendAuxiliaryCommandResponse**.

FAIL -

- None.

Validated Feature List: AuxiliaryCommands_IRLampAuto

Annex A Test for Appendix A

A.1 Required Number of Devices Summary

Required number of devices and Device feature dependency used in this test specification are listed in the Table.

Table A.1. Required Number of Devices Summary

Feature ID	Feature Name	Required Number of Devices	Check Condition based on Device Features	Check Condition based on Device Features ID
tc.UsernameToken	Username Token	1 (Note: Username Token feature shall be passed with at least one Device and can by not detected with other devices with supporting of WS-Username Token)	WS-Username Token	WSU
tc.HTTPDigest	HTTP Digest	3	Digest	Digest
tc.Capabilities	Capabilities	3	None	All
tc.GetServices	Get Services	3	GetServices is supported by Device.	GetServices
tc.GetServicesWithCapabilities	Get Services with Capabilities	1	GetServices is supported by Device.	GetServices
tc.EventHandling	Event Handling	3	None	All
tc.KeepAliveForPullPointEventHandling	Keep Alive for Pull Point Event Handling	3	None	All
tc.Discovery	Discovery	3	None	All
tc.NVTDiscoveryTypeFilter	Network Video Transmitter Discovery Type Filter	3	Network Video Transmitter Discovery Type is supported by Device.	DiscoveryTypesDnNetworkVideoTransmitter

Feature ID	Feature Name	Required Number of Devices	Check Condition based on Device Features	Check Condition based on Device Features ID
tc.DeviceDiscoveryTypeFilter	Device Discovery Type Filter	3	Device Discovery Type is supported by Device.	DiscoveryTypesTdsDevice
tc.NetworkConfiguration	Network Configuration	3	None	All
tc.System	System	3	None	All
tc.UserHandling	User Handling	3	None	All
tc.RelayOutputs	Relay Outputs	1	Relay Outputs (Device Management Service) is supported by Device.	DeviceIORelayOutputs
tc.NTP	NTP	1	NTP is supported by Device.	NTP
tc.DynamicDns	Dynamic DNS	1	Dynamic DNS is supported by Device.	DynamicDNS
tc.ZeroConfiguration	Zero Configuration	1	Zero Configuration is supported by Device.	ZeroConfiguration
tc.IPAddressFiltering	IP Address Filtering	1	IP Filter is supported by Device.	IPFilter
tc.PersistentNotificationStorageRetrieval	Persistent Notification Storage Retrieval	1	Persistent Notification Storage is supported by Device.	PersistentNotificationStorage
tc.SystemDateAndTimeConfiguration	System Date and Time Configuration	1	None	All
tc.HTTPFirmwareUpgrade	HTTP Firmware Upgrade	1	HTTP Firmware Upgrade is supported by Device.	HttpFirmware Upgrade

Feature ID	Feature Name	Required Number of Devices	Check Condition based on Device Features	Check Condition based on Device Features ID
tc.HTTPSystemBackup	HTTP System Backup	1	HTTP System Backup is supported by Device.	HttpSystemBackup
tc.HTTPSystemRestore	HTTP System Restore	1	HTTP System Backup is supported by Device.	HttpSystemBackup
tc.MonitoringNotifications	Monitoring Notifications	1	Monitoring/ProcessorUsage or Monitoring/OperatingTime/LastReset or Monitoring/OperatingTime/LastReboot or Monitoring/OperatingTime/LastClockSynchronization is supported by Device.	MonitoringProcessorUsageEvent OR MonitoringOperatingTimeLastResetEvent OR MonitoringOperatingTimeLastRebootEvent OR MonitoringOperatingTimeLastClockSynchronizationEvent
tc.DeviceManagementNotifications	Device Management Notifications	1	Check Condition based on Device Features: Device/HardwareFailure/FanFailure or Device/HardwareFailure/PowerSupplyFailure or Device/HardwareFailure/StorageFailure or Device/HardwareFailure/TemperatureCritical or Monitoring/Backup/Last is	MonitoringBackupLastEvent OR DeviceHardwareFailureFanFailureEvent OR DeviceHardwareFailurePowerSupplyFailureEvent OR DeviceHardwareFailureStorageFailureEvent OR DeviceHardwareFailureTemp

Feature ID	Feature Name	Required Number of Devices	Check Condition based on Device Features	Check Condition based on Device Features ID
			supported by Device.	eratureCriticalEvent
tc.HostnameConfiguration	Hostname Configuration	1	None	All
tc.DNSConfiguration	DNS Configuration	1	None	All
tc.NetworkProtocolsConfiguration	Network Protocols Configuration	1	None	All
tc.HTTPDigestForRTSP	HTTP Digest Authentication for RTSP	TO BE DISCUSSED	Profile T	ProfileTSupported
tc.AuxiliaryCommands	Auxiliary Commands	1	None (ONVIF Profile T Simulator is used as device).	None (ONVIF Profile T Simulator is used as device)