ONVIF™

Core Client Test Specification

Version 16.07

July 2016

REVISION HISTORY

Vers.	Date	Description
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, CAPABILITY-1, CAPABILITY-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 CAPABILITY-1 GET SERVICES test case has been updated. The CAPABILITY-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 CAPABILITY-1 GET SERVICES test case has been added. The CAPABILITY-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 Security

Security section defines security mechanism for two different authentication methods: Digest Authentication and Username Token Profile. The scope of this specification is limited to Message level security.

1.3 Capabilities

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

1.4 Get Services with Capabilities

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

1.5 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.

1.6 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.

1.7 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.

1.8 Network Configuration

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

1.9 System

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

1.10 User Handling

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

1.11 Relay Outputs

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

1.12 NTP

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

1.13 Dynamic DNS

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

1.14 Zero Configuration

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

1.15 IP Address Filtering

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

1.16 Persistent Notification Storage Retrieval

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

1.17 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.

1.18 HTTP Firmware Upgrade

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

1.19 HTTP System Backup

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

1.20 HTTP System Restore

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

1.21 Monitoring Notifications

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

1.22 Device Management Notifications

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

1.23 Hostname Configuration

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

1.24 DNS Configuration

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

1.25 Network Protocols Configuration

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

2 Normative references

- ONVIF Conformance Process Specification:

  http://www.onvif.org/Documents/Specifications.aspx

- ONVIF Profile Policy:

  http://www.onvif.org/Documents/Specifications.aspx

- ONVIF Core Specifications:

  http://www.onvif.org/Documents/Specifications.aspx

- ONVIF Core Client Test Specification:

  http://www.onvif.org/Documents/Specifications.aspx

- ONVIF Profile A Specification:

  http://www.onvif.org/Documents/Specifications.aspx

- ONVIF Access Rules Specification:

  http://www.onvif.org/Documents/Specifications.aspx

- ONVIF Credential Specification:

  http://www.onvif.org/Documents/Specifications.aspx

- ONVIF Schedule Specification:

  http://www.onvif.org/Documents/Specifications.aspx

- 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/

- ONVIF Streaming Specification:

  http://www.onvif.org/Documents/Specifications.aspx

- 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]
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
tns1	http://www.onvif.org/ver10/topics	The namespace for the ONVIF topic namespace

Prefix	Namespace URI	Description
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].

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 and feature requirement level for the Profiles, which will be used for Profiles conformance.

If Feature Level Requirement is defined as Mandatory for some Profile, Client shall pass Expected Scenario Under Test for each Device with this Profile support to claim this Profile Conformance.

If Feature Level Requirement is defined as Conditional, Optional for some Profile, Client shall pass Expected Scenario Under Test for at least one Device with this Profile support to claim feature as supported.

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 Security Test Cases

5.1 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).
   - IF Device returns a correct response, THEN Client determines that Device does not require any user authentication toward the command according to the configured security policy.
2. Client shall provide with the proper level of user credential to continue the test procedure in the following cases:
   - IF Device returns HTTP 401 Unauthorized error along with WWW-Authentication: Digest header, THEN Client determines that Device supports HTTP Digest authentication.
   - IF Device returns SOAP fault (Sender/NotAuthorized) message, THEN Client determines that UsernameToken is supported by Device.
3. Client is considered as supporting Security User Authentication if the following conditions are met:
   - Device returns a valid response to specific request with UsernameToken authentication header OR
   - Device returns a valid response to specific request with HTTP Digest authentication header.
4. Client is considered as NOT supporting Security (User Authentication) if the following is TRUE:
   - All HTTP Digest attempts detected are failed AND
   - All UsernameToken attempts detected are failed.

5.2 USER TOKEN PROFILE

Test Label: Security - User token profile

Test Case ID: SECURITY-1

Profile S Normative Reference: Mandatory

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Profile Q Normative Reference: None

Profile A 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: Security_UsernameToken

5.3 HTTP DIGEST AUTHENTICATION

Test Label: Security - HTTP Digest Authentication.

Test Case ID: SECURITY-2

Profile S Normative Reference: Mandatory

Profile G Normative Reference: Mandatory

Profile C Normative Reference: Mandatory

Profile Q Normative Reference: None

Profile A 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: Security_HTTPDigest

6 Capabilities Test Cases

6.1 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.

6.2 GET SERVICES

Test Label: Capabilities - Determine the available Services

Test Case ID: CAPABILITY-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

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_GetServices

6.3 GET CAPABILITIES

Test Label: Capabilities - Get Device Capabilities

Test Case ID: CAPABILITY-2

Profile S Normative Reference: Mandatory

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Profile Q Normative Reference: None

Profile A 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

7 Get Services with Capabilities Test Cases

7.1 Feature Level Requirement:

Validated Feature: get_services_capabilities

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile G Requirement: Optional

Profile Q Requirement: Optional

Profile S Requirement: None

7.2 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.

7.3 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

Profile S Normative Reference: None

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: get_services_capabilities.get_services

8 Event Handling Test Cases

8.1 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.

8.2 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

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

8.3 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

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_WS-BaseNotification

8.4 METADATA STREAMING

Test Label: Event Handling - Metadata Streaming

Test Case ID: EVENTHANDLING-3

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

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 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.

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 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] It contains **trt:MediaUri\tt: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

9 Keep Alive for Pull Point Event Handling Test Cases

9.1 Feature Level Requirement:

Validated Feature: keep_alive_pp_event_handling

Profile A Requirement: Mandatory

Profile C Requirement: Mandatory

Profile S Requirement: Conditional

Profile Q Requirement: Optional

Profile G Requirement: Conditional

9.2 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.

9.3 RENEW

Test Label: Advanced Pull Point Event Handling - Renew

Test Case ID: KEEPALIVEPPEVENTHANDLING-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

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: keep_alive_pp_event_handling.renew

9.4 PULL MESSAGES AS KEEP ALIVE

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

Test Case ID: KEEPALIVEPPEVENTHANDLING-2

Profile A Requirement: Mandatory

Profile C Requirement: Mandatory

Profile S Requirement: Conditional

Profile Q Requirement: Optional

Profile G Requirement: Conditional

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: keep_alive_pp_event_handling.pullmessages

10 Discovery Test Cases

10.1 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.

10.2 DISCOVERING DEVICES

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

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_WS-Discovery

11 Network Configuration Test Cases

11.1 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.

11.2 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

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

11.3 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

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

11.4 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

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

11.5 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

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

12 System Test Cases

12.1 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.

12.2 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

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

13 User Handling Test Cases

13.1 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 OR
   - No Valid Device Response to GetUsers request OR
   - No Valid Device Response to SetUser request OR
   - No Valid Device Response to DeleteUsers request.

13.2 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

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
  - [S6] Device response contains "HTTP/* 200 OK" AND
  - [S7] Device response contains "<CreateUsersResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: UserHandling_CreateUsers

13.3 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

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

13.4 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

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
  - [S5] Device response contains "HTTP/* 200 OK" AND
  - [S6] Device response contains "<SetUserResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: UserHandling_SetUser

13.5 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

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] Device response contains "HTTP/* 200 OK" AND
  - [S4] Device response contains "<DeleteUsersResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: UserHandling_DeleteUsers

14 Relay Outputs Test Cases

14.1 Expected Scenarios Under Test:

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

14.2 GET RELAY OUTPUTS

Test Label: Relay Outputs - GetRelayOutputs

Test Case ID: RELAYOUTPUTS-1

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Profile Q Normative Reference: None

Profile A Normative Reference: None

Feature Under Test: Relay Outputs

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

Pre-Requisite:

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

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

1. Client invokes GetRelayOutputs request message to get list of all available relay outputs and their settings.
2. Device responds with code HTTP 200 OK and GetRelayOutputsResponse message.

Test Result:

PASS -

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

FAIL -

- The Client failed PASS criteria.

Validated Feature List: RelayOutputs_GetRelayOutputs

14.3 SET RELAY OUTPUT STATE

Test Label: Relay Outputs - SetRelayOutputState

Test Case ID: RELAYOUTPUTS-2

Profile S Normative Reference: Conditional

Profile G Normative Reference: Optional

Profile C Normative Reference: Optional

Profile Q Normative Reference: None

Profile A Normative Reference: None

Feature Under Test: Relay Outputs

Test Purpose: To verify that Client is able to trigger relay output using the SetRelayOutputState operation.

Pre-Requisite:

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

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

1. Client invokes SetRelayOutputState request message to change state of relay output on Device.
2. Device responds with code HTTP 200 OK and SetRelayOutputStateResponse message.

Test Result:

PASS -

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

FAIL -

- The Client failed PASS criteria.

Validated Feature List: RelayOutputs_SetRelayOutputState

14.4 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

Profile Q Normative Reference: None

Profile A Normative Reference: None

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.

Pre-Requisite:

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

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

1. Client invokes SetRelayOutputSettings request message 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>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **SetRelayOutputSettings** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetRelayOutputSettings** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<SetRelayOutputSettings>" tag after the "<Body>" tag AND
  - [S2] "<SetRelayOutputSettings>" includes tag: "<RelayOutputToken>" with non-empty string value AND
  - [S4] "<Properties>" includes tag: "<Mode>" with "Bistable" value AND
  - [S7] Device response contains "HTTP/* 200 OK" AND
  - [S8] Device response contains "<SetRelayOutputSettingsResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: RelayOutputs_SetRelayOutputSettingsBistable

14.5 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

Profile Q Normative Reference: None

Profile A Normative Reference: None

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.

Pre-Requisite:

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

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

1. Client invokes SetRelayOutputSettings request message 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>" tag then Test shall be deemed as "NOT DETECTED".

PASS -

- Client **SetRelayOutputSettings** request messages are valid according to XML Schemas listed in Namespaces AND
- Client **SetRelayOutputSettings** request in Test Procedure fulfills the following requirements:
  - [S1] Client request contains "<SetRelayOutputSettings>" tag after the "<Body>" tag AND

- [S2] "<SetRelayOutputSettings>" includes tag: "<RelayOutputToken>" with non-empty string value AND
- [S4] "<Properties>" includes tag: "<Mode>" with "Monostable" value AND
- [S7] Device response contains "HTTP/* 200 OK" AND
- [S8] Device response contains "<SetRelayOutputSettingsResponse>" tag.

FAIL -

- The Client failed PASS criteria.

Validated Feature List: RelayOutputs_SetRelayOutputSettingsMonostable

15 NTP Test Cases

15.1 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.

15.2 GET NTP SETTINGS

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 A Normative Reference: None

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

15.3 SET NTP SETTINGS

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 A Normative Reference: None

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

16 Dynamic DNS Test Cases

16.1 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.

16.2 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

Profile Q Normative Reference: None

Profile A Normative Reference: None

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_GetDynamicDNS

16.3 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

Profile Q Normative Reference: None

Profile A Normative Reference: None

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_SetDynamicDNS

17 Zero Configuration Test Cases

17.1 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.

17.2 GET ZERO CONFIGURATION SETTINGS

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

Profile A Normative Reference: None

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

17.3 SET ZERO CONFIGURATION SETTINGS

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

Profile A Normative Reference: None

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

18 IP Address Filtering Test Cases

18.1 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.

18.2 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 Q Normative Reference: None

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

18.3 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 Q Normative Reference: None

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

18.4 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 Q Normative Reference: None

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

18.5 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 Q Normative Reference: None

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

18.6 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 Q Normative Reference: None

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

18.7 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 Q Normative Reference: None

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

18.8 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 Q Normative Reference: None

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

19 Persistent Notification Storage Retrieval Test Cases

19.1 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.

19.2 SEEK STORED EVENTS IN DEVICE

Test Label: Persistent Notification Storage Retrieval - Seek

Test Case ID: PERSISTENTNOTIFICATIONSTORAGERETRIEVAL-1

Profile S Normative Reference: Optional

Profile G Normative Reference: Optional

Profile C Normative Reference: Conditional

Profile Q Normative Reference: None

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

20 System Date and Time Configuration Test Cases

20.1 Feature Level Requirement:

Validated Feature: system_date_and_time_configuration

Profile A Requirement: Conditional

Profile C Requirement: Optional

Profile G Requirement: Optional

Profile Q Requirement: Conditional

Profile S Requirement: Optional

20.2 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.

20.3 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: system_date_and_time_configuration.get_system_date_and_time

20.4 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: system_date_and_time_configuration.set_system_date_and_time

21 HTTP Firmware Upgrade Test Cases

21.1 Feature Level Requirement:

Validated Feature: http_firmware_upgrade

Profile A Requirement: None

Profile C Requirement: None

Profile G Requirement: None

Profile Q Requirement: Conditional

Profile S Requirement: None

21.2 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.

21.3 FIRMWARE UPGRADE VIA HTTP

Test Label: Firmware Upgrade via HTTP - Start Firmware Upgrade

Test Case ID: HTTPFIRMWAREUPGRADE-1

Profile A Normative Reference: None

Profile C Normative Reference: None

Profile G Normative Reference: None

Profile Q Normative Reference: Conditional

Profile S Normative Reference: None

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: http_firmware_upgrade.http_firmware_upgrade

22 HTTP System Backup Test Cases

22.1 Feature Level Requirement:

Validated Feature: http_system_backup

Profile A Requirement: None

Profile C Requirement: None

Profile G Requirement: None

Profile Q Requirement: Conditional

Profile S Requirement: None

22.2 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.

22.3 HTTP SYSTEM BACKUP

Test Label: System Backup via HTTP - Get System Uris

Test Case ID: HTTPSYSTEMBACKUP-1

Profile A Normative Reference: None

Profile C Normative Reference: None

Profile G Normative Reference: None

Profile Q Normative Reference: Conditional

Profile S Normative Reference: None

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: http_system_backup.http_system_backup

23 HTTP System Restore Test Cases

23.1 Feature Level Requirement:

Validated Feature: http_system_restore

Profile A Requirement: None

Profile C Requirement: None

Profile G Requirement: None

Profile Q Requirement: Conditional

Profile S Requirement: None

23.2 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.

23.3 HTTP SYSTEM RESTORE

Test Label: System Restore via HTTP - Start System Restore

Test Case ID: HTTPSYSTEMRESTORE-1

Profile A Normative Reference: None

Profile C Normative Reference: None

Profile G Normative Reference: None

Profile Q Normative Reference: Conditional

Profile S Normative Reference: None

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: http_system_restore.http_system_restore

24 Monitoring Notifications Test Cases

24.1 Feature Level Requirement:

Validated Feature: monitoring_notifications

Profile A Requirement: None

Profile C Requirement: None

Profile G Requirement: None

Profile Q Requirement: Conditional

Profile S Requirement: None

24.2 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.

25 Device Management Notifications Test Cases

25.1 Feature Level Requirement:

Validated Feature: device_management_notifications

Profile A Requirement: None

Profile C Requirement: None

Profile G Requirement: None

Profile Q Requirement: Conditional

Profile S Requirement: None

25.2 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

26 Hostname Configuration Test Cases

26.1 Feature Level Requirement:

Validated Feature: hostname_configuration

Profile A Requirement: Optional

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 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.

26.3 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: hostname_configuration.get_hostname

26.4 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: hostname_configuration.set_hostname

27 DNS Configuration Test Cases

27.1 Feature Level Requirement:

Validated Feature: dns_configuration

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile G Requirement: Optional

Profile Q Requirement: Conditional

Profile S Requirement: Optional

27.2 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.

27.3 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: dns_configuration.get_dns

27.4 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: dns_configuration.set_dns

28 Network Protocols Configuration Test Cases

28.1 Feature Level Requirement:

Validated Feature: network_protocols_configuration

Profile A Requirement: Optional

Profile C Requirement: Optional

Profile G Requirement: Optional

Profile Q Requirement: Conditional

Profile S Requirement: Optional

28.2 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.

28.3 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: network_protocols_configuration.get_network_protocols

28.4 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: network_protocols_configuration.set_network_protocols