Status: Approved

Version: 1.2

SunSpec Common Smart Inverter Profile (CSIP) Conformance Test Procedures

SunSpec Test Procedures

Abstract

This document provides conformance test procedures for compliance with the requirements specified in the *Common Smart Inverter Profile*.

License Agreement and Copyright Notice

Prepared by the SunSpec Alliance

4040 Moorpark Avenue, Suite 110

San Jose, CA 95117

Website: SunSpec Alliance

Email: info@sunspec.org

Revision History

Version	Date	Comments
0.9	01-31-2018	Draft version
1.0	05-31-2018	Incorporates community review comments and corrects typographical and cross-reference errors
1.2	07-24-2019	Moved test requirements tracking to a single section. Removed test prerequisite sections. Removed requirement for testing with HTTP. Changed the Server [S], Client [C] terminology to Server [S], Client [C], Test Server [TS], Test Client [TC]. Removed redundant tests: CORE-007, CORE-008, CORE-020. Updated test criteria to clarify which system types for which each test is applicable and the role of each test element.

About the SunSpec Alliance

The SunSpec Alliance is a trade alliance of developers, manufacturers, operators, and service providers together pursuing open information standards for the distributed energy industry. SunSpec standards address most operational aspects of PV, storage, and other distributed energy power plants on the smart grid, including residential, commercial, and utility-scale systems, thus reducing cost, promoting innovation, and accelerating industry growth.

Over 100 organizations are members of the SunSpec Alliance, including global leaders from Asia, Europe, and North America. Membership is open to corporations, nonprofits, and individuals. For more information about the SunSpec Alliance, or to download SunSpec specifications at no charge, visit sunspec.org.

About the SunSpec Specification Process

SunSpec Alliance specifications are initiated by SunSpec members to establish an industry standard for mutual benefit. Any SunSpec member can propose a technical work item. Given sufficient interest and time to participate, and barring significant objections, a workgroup is formed, and its charter is approved by the board of directors. The workgroup meets regularly to advance the agenda of the team.

The output of the workgroup is generally in the form of a SunSpec Interoperability Specification. These documents are considered to be normative, meaning there is a matter of conformance required to support interoperability. The revision and associated process of managing these documents is tightly controlled. Other documents are informative, or make some recommendation with regard to best practices, but are not a matter of conformance. Informative documents can be revised more freely and more frequently to improve the quality and quantity of information provided.

SunSpec Interoperability Specifications follow a lifecycle pattern of: DRAFT, TEST, APPROVED, and SUPERSEDED.

For more information or to download a SunSpec Alliance specification, go to https://sunspec.org/about-sunspec-specifications/.

Acknowledgements

The following contributed to this specification:

Bob Fox, SunSpec Alliance

Gene Hartsell, SunSpec Alliance

Steve Kang, QualityLogic, Inc.

Gordon Lum, Kitu Systems, Inc.

James Mater, QualityLogic, Inc.

Tom Tansy, SunSpec Alliance

Contents

Appendix

Figures

1 Introduction

This document specifies test procedures for validating conformance to Common Smart Inverter Profile (CSIP) testable requirements for server, Distributed Energy Resource (DER) client, and DER aggregator client implementations.

1.1 Purpose

The objective of these test procedures is to verify that the implementation is compliant with the IEEE 2030.5 functionality and options as specified in the CSIP document.

1.2 Scope and Applicability

The CSIP documentation specifies a usage model for communicating with smart inverter systems that implement the IEEE 2030.5 standard, and imposes the following IEEE 2030.5 functional qualifications:

- Identifies the required IEEE 2030.5 functional subset that must be supported.
- Where applicable, specifies the required functionality option that must be used to promote interoperability.
- Identifies capacity constraints, which might differ from the default values specified in IEEE 2030.5.
- This specification is currently based on versions of the CSIP specification and IEEE 2030.5 specification that are undergoing review and updating. It is expected that updates to this specification will be required after updated versions of those specifications are released.

1.3 Overview

CSIP specifies that IEEE 2030.5 shall be the default protocol for communicating with a utility server.

IEEE 2030.5 standardizes communication between smart grid-enabled components. The standard uses current networking technologies to provide a secure and interoperable framework for information exchange.

CSIP specifies a usage model for communicating with systems having smart inverters implementing the IEEE 2030.5 standard, which supports functionality for a broad range of smart devices.

1.3.1 Client-server Model

The IEEE 2030.5 standard specifies a REST architecture. The HTTP protocol is a required baseline for interoperable IEEE 2030.5 servers and clients, which must comply with the IETF RFC 2616 standard.

Tests specified in this document are structured to describe both the server and client behavior and can be used to test either a server or client implementation.

1.3.2 CSIP Client-server Framework

The CSIP specification describes the interaction between a utility server and a DER system in two ways:

- direct DER communications, including generating facility management system (GFEMS)
- aggregator mediated communications

Accordingly, the test specifications address the following functional profiles:

- server tests
- DER client tests
- DER aggregator client tests

Server tests consist of most of the tests specified in the test procedures. A server implementation must be able to support interaction with a DER client or DER aggregator.

DER client tests include the basic IEEE 2030.5 and DER functionality required for client testing.

Additional DER aggregator client tests include functionality required for aggregator-mediated communications.

The specific tests associated with each of the three profiles are provided in the Profile Test Conformance section.

1.4 References

Power Line Communications Committee of the IEEE Communications Society. "P2030.5™/D2 Draft Standard for Smart Energy Profile Application Protocol." March 2018.

Common Smart Inverter Profile Working Group. "Common Smart Inverter Profile IEEE 2030.5 Implementation Guide for Smart Inverters, Version 2.1." March 2018.

Fielding, R., Gettys, J., Mogul, J. Frystyk, H., Masinter, L., Leach, P., Berners-Lee, T., "Hypertext Transfer Protocol -- HTTP/1.1." RFC 2616, June 1999.

Rescorla, E., "HTTP Over TLS." RFC 2818, May 2000.

Hadley, M.J., "Web Application Description Language (WADL)." 2009.

2 Nomenclature

In this document, the word *shall* is used to indicate a mandatory requirement. The word *should* is used to indicate a recommendation. The word *may* is used to indicate a permissible action. The word *can* is used for statements of possibility and capability.

2.1 Terms

Term	Meaning
Aggregator	A system that acts as an intermediary to multiple DER systems.
Client	An implementation that performs as a DER or DER aggregator.
Device	A physical object that performs a set of functions. Examples are inverters, trackers, and modules.
Server	A system that interacts with DER clients and/or DER aggregator clients.

2.2 Acronyms and Abbreviations

Acronym	Meaning
CA	Certificate Authority
CSIP	Common Smart Inverter Profile
DDERC	Default DER Control
DER	Distributed Energy Resource
DERC	DER Control
DERP	DER Program
EUT	Equipment Under Test
EXI	Efficient XML Interchange
GFEMS	Generating Facility Energy Management System
IETF	Internet Engineering Task Force
LFDI	Long Form Device Identifier
MCA	Manufacturer's CA
MICA	Manufacturing Issuing CA
OOB	Out-of-band
PEV	Plug-in Electric Vehicle
PF	Power Factor
RFC	IETF Request for Comments
SERCA	Smart Energy Root CA
SFDI/sFDI	Short Form Device Identifier
TLS	Transport Layer Security
UDP	User Datagram Protocol
URI	Uniform Resource Identifier
WADL	Web Application Description Language
XML	Extensible Markup Language

3 Testing Overview

3.1 IEEE 2030.5 Test Scope

The tests in this specification cover IEEE 2030.5 functionality, as addressed in the CSIP requirements.

The IEEE 2030.5 requirements in this specification are derived directly from the functionality specified as mandatory in the IEEE 2030.5 specification.

IEEE 2030.5 functional areas are categorized, and each requirement is assigned an ID, which consists of the function category prefix and a unique, sequential number. For example, requirement ID `DER.001` identifies requirement number one of the distributed energy resources (DER) function set. See the appendix for the list of testable IEEE 2030.5 requirements.

The following table lists the requirement categories, ID prefix, and IEEE 2030.5 section reference. Shaded table rows indicate categories outside the test scope according to CSIP requirements. Categories outside of the test scope are not included in this specification.

Category	Description	IEEE 2030.5 Sections
BASE	Base functionality used by multiple function sets	8, B.1.2, B.1.5, B1.6
BILL	Billing function set	10.6, B.1.18
CFG	Configuration function set	9.6, B.1.12
DER	Distributed energy resources function set	10.9, B.1.21
DI	Device information function set	9.2, B.1.8
DNS	Discovery	7
DRLC	Demand response and load control function set	10.2, B.1.14
EVENT	Event rules	10.1.3, B.1.2.2
FILE	File Download function set	9.7, B.1.13
FLOW	Flow reservation function set	10.8, B.1.20
GEN	General networking	4, 9, 11
LOG	Log Event function set	9.5, B.1.11
METER	Metering function set	10.3, B.1.15
MSG	Messaging function set	10.5, B.1.17
MULTI	Multi-Server	10.1.5
MUP	Metering mirror function set	10.10, B.1.15.1
NSTAT	Network status	9.4, B.1.10
POWER	Power status function set	9.3, B.1.9
PPAY	Prepayment function set	10.7, B.1.19
PRICE	Pricing function set	10.4, B.1.16
RAND	Randomization	10.1.4
SEC	Security	6
TIME	Time function set	9.1, B.1.7

3.2 Testing Assumptions and Notation

A client (Client) or server (Server) can each be tested for IEEE 2030.5 and CSIP compliance using this test plan.

The implementation being validated performs either the Client or Server functionality. The testing framework provides the corresponding testing implementation.

3.2.1 General Communication and Payload Requirements

The following general communication network requirements shall apply.

- In the test descriptions, all HTTP communication must be done using HTTP with TLS (HTTPS).
- All tests assume IPv6 or IPv4.
- All tests assume an XML payload unless otherwise specified.
- A packet sniffer, such as WireShark or equivalent, is recommended for independent network traffic analysis between the Client and Server.

When testing a client implementation, a Server is a connection endpoint. When testing a server implementation, a Client is a connection endpoint.

3.2.2 Mandatory Conformance Requirements for All Tests

The following requirements shall apply to all tests unless the test description specifically waives the requirement.

- [GEN.001] IEEE 2030.5 servers and clients SHALL be compliant with [RFC 7230], [RFC 7231], [RFC 7232], [RFC 21 7233], [RFC 7234], and [RFC 7235].].
- [GEN.003] Content shall be transferred using either of the following content types:
    - application/sep+xml
    - application/sep-exi
- [GEN.004] All resources SHALL contain links to their subordinate resources to support flexibility in URIs and future extensibility.
- [GEN.005] Hexadecimal values SHALL be represented with one leading zero, if needed, to ensure an even number of digits.
- [GEN.006] IEEE 2030.5 devices SHALL conform to the interface specifications contained in the WADL.
- [GEN.007] Devices SHALL conform to the WADL specification as per [WADL].
- [GEN.008] Devices SHALL conform to the WADL definition in [IEEE 2030.5 SM].
- [GEN.009] By implication, all resource representations SHALL validate per the schema [IEEE 2030.5 SM] within the standardized IEEE 2030.5 XML namespace (http://ieee.org/2030.5).
- [GEN.010] Resources located at URIs returned in the href attribute of "Link" specializations (e.g., EndDeviceListLink, SelfDeviceLink) SHALL conform to the schema definition for that object.

- [GEN.011] If a client PUTs or POSTs a resource to a server containing attributes or elements that instead are to be populated by the server (e.g., href), the server SHALL return an HTTP 400 error.
- [GEN.012] If a function set is not implemented, link elements to resources in that function set SHALL NOT be included.
- [GEN.013] URIs SHALL NOT be greater than 255 bytes in length. In practice, URIs SHOULD be much smaller than 80 bytes.
- [GEN.014] List resource elements SHALL be ordered according to this specified list ordering.
- [GEN.015] The first ordinal position of the list SHALL be designated with a value of '0' and the maximum possible value is “4294967295.”.
- [GEN.016] If this query string parameter is not specified, the default start value SHALL be '0'.
- [GEN.017] The parameter SHALL be ignored if the primary key is not time-based.
- [GEN.018] The format of the parameter SHALL be a 64-bit decimal number with identical semantics as that of the TimeType (see Section 10.2 and the XML XSD in [ZB 13-0201]).
- [GEN.019] If this query string parameter is not specified, the default limit SHALL be '1'.
- [GEN.020] If both a ""start"" and ""after"" query string parameter are used simultaneously, the ""after"" query string parameter SHALL have precedence.
- [GEN.021] ""start"" position 0 SHALL be relative to the position specified by the ""after"" parameter.
- [GEN.022] If a query string requests a list element that does not exist (e.g., s=3 when there are two items in the list), servers SHALL return an empty list representation.
- [GEN.023] If a particular query string parameter appears more than once, then the first occurrence of the query string parameter SHALL be used (in left-to-right order) and subsequent occurrences MUST be ignored.
- [GEN.024] Server receipt of a query parameter unknown to the server MUST be ignored by the server and MUST NOT generate an HTTP error.
- [GEN.025] Servers MUST NOT generate resource representations containing href attributes that contain query parameters.
- [GEN.026] Clients MUST ignore query parameters contained in resource hrefs, but SHOULD NOT remove them if the URI is used for subsequent RESTful exchanges.
- [GEN.027] If an empty list representation is requested, either through the use of query string parameters such as l=0 or when the list itself is empty, the server SHALL return no subordinate representations, but SHALL return any other elements that may be defined for the list.
- [GEN.028] Clients MUST NOT assume any index semantics for list URIs.
- [GEN.030] List resources SHALL support ""start"" and ""limit"" query string parameters, thus always supporting paging.

- [GEN.031] List resources that have a time-based primary key SHALL support the ""after"" query string parameters.
- [GEN.032] All subordinate resources of list resources SHALL include an href attribute containing the URI of the subordinate resource.
- [GEN.033] When queried, list resources SHALL return subordinate resources in the order defined by the List Ordering.
- [GEN.034] All subordinate resources of list resources that support multiple types, e.g., NotificationList, SHALL include an xsi:type attribute.
- [GEN.035] In this case, the XML Schema Instance Namespace must also be declared.
- [GEN.036] Non-list resources SHALL NOT support the defined query string parameters.
- [GEN.037] However, IEEE 2030.5 clients may encounter any HTTP response code defined by [RFC 2616] and, all use of, and response to HTTP response codes SHALL be specification and RFC compliant.
- [GEN.038] A client MUST be prepared to accept one or more 1xx status responses prior to a regular response, even if the client does not expect a 100 (Continue) status message.
- [GEN.039] The Location header SHALL be used in conjunction with this response code to indicate the URI of the newly created resource.
- [GEN.040] If a new resource is created, the origin server MUST inform the user agent via the 201 (Created) response.
- [GEN.041] Note that [RFC 2616] requires the Content-Range and Date headers MUST be present in the response.
- [GEN.042] The Location header SHALL be used in conjunction with this response code to indicate the new URI of the requested resource.
- [GEN.043] All Internet-based HTTP/1.1 servers MUST respond with a 400 (Bad Request) status code to any HTTP/1.1 request message which lacks a Host header field.
- [GEN.044] The response MUST include a WWW-Authenticated header field containing a challenge applicable to the requested resource.
- [GEN.045] The response MUST include an Allow header containing a list of valid methods for the requested resource.
- [GEN.046] A response with status code 206 (Partial Content) MUST NOT include a Content-Range field with a byte-range-resp-spec of "*".
- [GEN.047] If a server receives a request containing an Expect field that includes an expectation-extension that it does not support, it MUST respond with a 417 (Expectation Failed) status.
- [GEN.048] The recipient of the entity MUST NOT ignore any Content - (e.g., Content-Range) headers that it does not understand or implement and MUST return a 501 (Not Implemented) response in such cases.

- [GEN.049] If a client wants to operate with minimal understanding of HTTP response codes, it needs only to examine the first digit of the response code to understand the general category of the response and "treat any unrecognized response as being equivalent to the x00 status code of that class, with the exception that an unrecognized response MUST NOT be cached."
- [GEN.050] Application payload message encoding using both XML [XML] and EXI [EXI] SHALL be supported by all servers.
- [GEN.051] Application payload message encoding using either XML [XML] or EXI [EXI] SHALL be supported by all clients.
- [GEN.052] The XML version number shall be 1.0.
- [GEN.053] For XML payloads, the encoding SHALL be UTF-8.
- [GEN.056] A client SHALL declare acceptable media types using the HTTP Accept header.
- [SEC.009] Resource access requiring application layer authentication, data confidentiality and integrity checking SHALL occur through requests from a client to the server using HTTP over TLS [RFC 2818] (also known as HTTPS) using TLS version 1.2 [RFC 5246].
- [SEC.014] The use of TLS [RFC 5246] requires that all hosts implementing server functionality SHALL use a Device Certificate whereby the server presents its Device Certificate as part of the TLS handshake.

3.2.3 Server [S]

The Server is the set of equipment needed to implement an IEEE 2030.5- and CSIP-compliant server device. Minimally, the Server should be able to create the required server resources to run each test.

For the tests described in this document, the Server configuration assumes the following parameters:

- IP Address = `local IP address`
- TCP Port = `443 (or compatible TLS port)`
- Scheme = `HTTPS`
- Path to the `DeviceCapability` resource = `/dcap`
- Registration PIN = `111115`

3.2.4 Test Server [TS]

A Test Server is a server that is being used to provide server functionality and is not the equipment under test. In order to test a client more comprehensively, the test procedures may require a Test Server to be set up and perform operations that a production server may not always perform in normal operation.

3.2.5 Client [C]

The Client is the set of equipment needed to implement an IEEE 2030.5- and CSIP-compliant client device, such as an individual DER client or an aggregator. Minimally, the Client can initiate, retrieve, process, and act on DER server resources.

For the tests described in this document, the Client configuration assumes the following parameters:

- Test Server IP Address
- Test Server TCP Port
- Test Server scheme = `HTTPS`
- Path to the `DeviceCapability` resource
- Registration PIN = `111115`

On power-up or restart, the Client should do an HTTP GET of the `DeviceCapability` resource using the Test Server parameters.

3.2.6 Test Client [TC]

A test client is a client that is being used to provide client functionality and is not the equipment under test. In order to test a server more comprehensively, the test procedures may require a test client to be set up and perform operations that a production client may not always perform in normal operation. Normally, when a client is the equipment under test, its configuration or behavior would not be manipulated during testing. A test client, however, is manipulated during testing to test a range of conditions associated with the server functionality.

3.2.7 Equipment Under Test

Equipment under test (EUT) can be either:

- an IEEE 2030.5-compliant CSIP client, which can be either:
    - a DER inverter
    - an aggregator that acts on inverter control signals from the DER head end system
- a server device, which can be either:
    - a DER head end system that generates DER inverter control signals targeting the downstream aggregators
    - a DER system

3.3 Format of Test Description

Each test specification includes the following sections, which fully describes each test and how to do the test.

The test title includes an indication of which category for which the test is required: Device Client **[C]**, Aggregator Client **[A]**, and Server **[S]**. In the test descriptions, Device Clients and Aggregator Clients are both represented with the designation as a Client **[C]**.

Purpose

The purpose section describes test objectives.

Setup

The setup section describes client and server actions that must be done before performing the test.

Procedure

The procedure section lists the steps required to successfully complete the test. The steps required to be done in the specified order.

The **[S]** notation indicates server actions or events, which operate on a client.

The **[C]** notation indicates client actions or events, which operate on a server.

The **[TS]** notation indicates server actions or events associated with a test server.

The **[TC]** notation indicates server actions or events associated with a test client.

The **[T]** notation indicates the actions of the element being used to provide the testing functionality (either a test server or test client).

The **[U]** notation indicates the actions of the element being tested functionality (either a server or client).

Pass/Fail Criteria

The pass/fail criteria section lists the observable conditions for classifying a test as passed (successful completion) or failed (unsuccessful completion).

4 Profile Test Conformance

This section specified the required tests associated with the three testing profiles: Server, DER Client, and DER Aggregator Client.

Tests marked with an X in the following table are required for a conforming implementation for each profile.

Test ID	Server	DER Client	DER Aggregator Client
AGG-001	X		X
AGG-002	X		X
AGG-003	X		X
AGG-004	X		X
AGG-005	X		X
AGG-006	X		X
AGG-007	X		X
AGG-008	X		X
AGG-009	X		X
AGG-010	X		X
AGG-011	X		X
AGG-012	X		X
BASIC-001	X	X	X
BASIC-002	X	X	X
BASIC-003	X	X	X
BASIC-004	X	X	X
BASIC-005	X	X	X
BASIC-006	X	X	X
BASIC-007	X	X	X
BASIC-008	X	X	X
BASIC-009	X	X	X
BASIC-010	X	X	X
BASIC-011	X	X	X
BASIC-012	X	X	X
BASIC-013			
BASIC-014			
BASIC-015	X	X	X
BASIC-016	X	X	X
BASIC-017	X	X	X
BASIC-018	X	X	X
BASIC-019	X	X	X
BASIC-020	X	X	X
BASIC-021	X	X	X
BASIC-022	X	X	X
BASIC-023	X	X	X
BASIC-024	X	X	X
BASIC-025	X	X	X

Test ID	Server	DER Client	DER Aggregator Client
BASIC-026	X	X	X
BASIC-027	X	X	X
BASIC-028	X	X	X
BASIC-029	X	X	X
COMM-001			
COMM-002	X	X	X
COMM-003	X	X	X
COMM-004	X	X	X
CORE-001	X		
CORE-002	X		
CORE-003		X	X
CORE-004	X		
CORE-005	X	X	X
CORE-006	X		
CORE-009	X	X	X
CORE-010	X	X	X
CORE-011	X	X	X
CORE-012	X	X	X
CORE-013	X	X	X
CORE-014	X	X	X
CORE-018	X		X
CORE-019	X		X
CORE-021	X	X	X
CORE-022	X	X	X
ERR-001		X	X
ERR-002	X		X
MAINT-001	X		X
MAINT-002	X		X
MAINT-003	X		X
MAINT-004	X		X
MAINT-005	X		X
MAINT-006	X		X
UTIL-001	X		
UTIL-002	X		X
UTIL-003	X		X
UTIL-004	X		X

5 Communication Fundamentals Tests

This section specifies communication tests, which cover the following CSIP, section 3.3, functionality:

- TLS
- HTTP/HTTPS
- IPv6/IPv4
- Authentication
- Access control list
- Authorization
- Certificate fingerprints

The tests in this section ensure that client and server implementations conform to fundamental communication requirements and are a prerequisite for tests in subsequent sections of this document.

COMM-001 - Basic Discovery (xmDNS/DNS-SD)

Purpose

This test is optional for all device types.

The basic discovery xmDNS/DNS-SD test verifies the Client can correctly discover a local network IEEE 2030.5 server using xmDNS/DNS-SD and can communicate with the server using published server information. This test requires a network that routes xmDNS/DNS-SD packets from/to IEEE 2030.5 clients and servers.

Setup

1. Server supports xmDNS/DNS-SD queries for IEEE 2030.5 related services.
2. Server has configured its xmDNS/DNS-SD services to respond to main service and subtype queries to include all `DeviceCapability` resources.
3. Client can send xmDNS/DNS-SD queries and receiving Server responses.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Send an xmDNS/DNS-SD subtype query for any function set supported by the Server. For example, `derp.sub.smartenergy.tcp.site`. with QU bit off (off=multi-cast response for xmDNS).
3. **[S]** Receive the incoming xmDNS/DNS-SD query from Client and respond with a multi-cast response with PTR record, which has details about the resource for which the subtype query was sent for.
4. **[C]** Process the received xmDNS/DNS-SD response for IP address, port number, and path URI for the resource. For example, `derp`. Using the IP address, port number, and path URI information, perform an HTTP GET operation to retrieve the resource information from the Server.
5. **[S]** Receive the incoming HTTP GET request and send back the requested resource information as the HTTP GET response.
6. Repeat steps 1-4 but with QU bit on (on=unicast response for xmDNS).
7. Repeat steps 1-4, with QU bit off (for xmDNS), and use a different function set. For example, `tm.sub.smartenergy.tcp.site`.
8. Repeat steps 1-4, with QU bit off (for xmDNS), and request the main service query. For example, `smartenergy.tcp.site`.

Pass/Fail Criteria

- **[C]** Verify using the packet sniffer the xmDNS subtype query sent by the Client conforms to the xmDNS/DNS-SD protocol standard, including the QU bit and subtype name which is being requested. For example, `derp.sub.smartenergy.tcp.site`.
- **[S]** Verify using the packet sniffer that the Server response to the xmDNS subtype query was conformant to the xmDNS/DNS-SD protocol standard, including verification the

response was sent multi-cast to FF05::FB or 239.255.255.251 (IPV6 or IPV4 multi-cast destination). Confirm the Server xmDNS response had the required PTR/A/AAA/SRV record/values.

- **[C]** Verify the Client successfully received the xmDNS/DNS-SD response for its subtype query and issued an HTTP GET to the URL in the xmDNS/DNS-SD response.
- **[S]** Verify the Server successfully received the HTTP GET request from the Client and sent back a response with the resource payload specific to fulfill the HTTP GET request.
- Repeat the test steps 1-4 and evaluate the steps 1-4 Pass Criteria except the xmDNS response shall be unicast, instead of multi-cast.
- Repeat the test steps 1-4 and evaluate the steps 1-4 Pass Criteria. This sequence of tests shall use multi-cast because QU bit if OFF.
- Repeat the test steps 1-4 and evaluate the steps 1-4 Pass Criteria. This sequence of tests shall use multi-cast because QU bit if OFF.

COMM-002 - Basic Discovery (Out-of-Band) [C, A, S]

Purpose

Verify ability for client to connect to server using out-of-band (OOB) configured IP address, port number, and dcap path.

Setup

1. Server network/dcap related configuration (IP address, port number and dcap location) is known and available.
2. Client can configure its connection to use the Server network/dcap information.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Configure the network/dcap related information to use the Server information and perform an HTTP GET operation on the DeviceCapability resource.
3. **[S]** Receive the incoming DeviceCapability HTTP GET request and respond back with the appropriate resource and response.
4. **[C]** Process the DeviceCapability response form the Server and perform additional HTTP GETs on one of the found resources.
5. **[S]** Receive the incoming HTTP GET request and send back the requested resource information as the HTTP GET response.

Pass/Fail Criteria

- **[C]** Client successfully uses the correct network/dcap information that is associated with the Server.
- **[C]** Client successfully issues HTTP GET on the `DeviceCapability` using the Server network/dcap information.
- **[S]** Server successfully responds to each of the HTTP GET request from the Client and returns the correct HTTP response and codes.
- **[C]** Client successfully issues additional HTTP GETs on one of the resources found in the `DeviceCapability` resource.

COMM-003 - Basic Security [C,A,S]

Purpose

Verify ability to connect to server using HTTPS and IEEE 2030.5 permissible cypher suite.

The basic security test verifies that the Client can correctly communicate with an IEEE 2030.5 server using basic security requirements. For example, the HTTPS, TLS 1.2, `TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8` cipher suite. TLS authentications are tested based on requirements specified in the IEEE 2030.5 Application Protocol Specification.

Setup

1. Server and Client support the TLS based HTTP communication as specified in the Requirements, including the use of mandatory `TLS_ECDHE_ECDSA_WITH _AES_128_CCM_8` cipher suite.
2. Server is configured to use either the default TLS port (443) or another port. Client is configured to use the supported TLS port and IP address from the Server.
3. Client can send and receive TLS based HTTPS messages as specified in the requirements, including the use of mandatory `TLS_ECDHE_ECDSA_WITH _AES_128_CCM_8` cipher suite.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Using the known IP address, port number, and `DeviceCapability` URI, send a TLS based HTTP GET request to the Server.
3. **[S]** Successfully receive the TLS based HTTP GET request and respond with the `DeviceCapability` resource payload through the TLS port number.

Pass/Fail Criteria

- **[C]** The Client successfully established a TLS HTTP session by conforming to the requirements specified in RFC 5246, section 7.4. Verify by inspecting the TLS packets, including verification that `TLS_ECDHE_ECDSA_WITH _AES_128_CCM_8` cipher suite was used. Successfully sent a TLS based HTTP GET request to the Server `DeviceCapability` resource using the known IP address, port number, and `DeviceCapability` URI.
- **[S]** Server successfully established a TLS HTTP session by conforming to the requirements specified in RFC 5246, section 7.4. Verify by inspecting the TLS packets, including verification that `TLS_ECDHE_ECDSA_WITH _AES_128_CCM_8` cipher suite was used. Successfully received the TLS based HTTP GET request and responded with the `DeviceCapability` resource payload as the HTTP GET response.

COMM-004 - Advanced Security [C, A, S]

Purpose

Verify ability to detect errors in certificate chain of peer and reject the connection.

The advanced security test verifies that the Client can communicate with the IEEE 2030.5 server using basic TLS/security requirements and can also handle more challenging requirements, including invalid scenarios. For example, handling broken connections, invalid certificates, and invalid root CA.

Setup

1. Server and Client support the TLS based HTTP communication as specified in the Requirements, including the use of mandatory `TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8` cipher suite and send TLS Alerts for error situations.
2. Server is configured to use either the default TLS port (443) or another port. Client is configured to use the supported TLS port and IP address from the Server.
3. Client can send and receive TLS based HTTPS messages as specified in the requirements, including the use of mandatory `TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8` cipher suite and send TLS Alerts for error situations.
4. Server and Client support two, three, and four chain length TLS certs where:
    - Certificate chain length two: SERCA -> Device Certificate
    - Certificate chain length three: SERCA->MICA->Device Certificate
    - Certificate chain length four: SERCA->MCA->MICA->Device Certificate

    Refer to the IEEE 2030.5 standard, Certificate Management section.
5. Additional TLS certs with following attributes:
    - Invalid MICA Extended Key Critical value
    - Invalid MICA Name Non-Critical Value
    - Invalid MICA Policy Mapping Non-Critical value
    - Self-signed device certificate

Procedure

1. **[T]** Record the Client/Server communications.
2. **[T]** Configure with TLS cert, chain length two: SERCA->Device Certificate, to establish a new TLS session.
3. **[C]** Using the known IP address, port number, and `DeviceCapability` URI, send a TLS based HTTP GET request to the Server.
4. **[S]** Successfully receive the TLS based HTTP GET request and respond with the `DeviceCapability` resource payload through the TLS port number.
5. **[T]** Configure with TLS cert, chain length three: SERCA->MICA->Device Certificate, and start a new TLS session establishment and repeat test steps 2 and 3.

6. **[T]** Configure with TLS cert, chain length four: SERCA->MCA->MICA->Device certificate and start a new TLS session establishment and repeat test steps 2 and 3.
7. **[T]** Configure with TLS cert, Invalid MICA Extended Key Critical, and stat a new TLS session establishment and repeat test steps 2 and 3.
8. **[T]** Configure with TLS cert, Invalid MICA Name Non-Critical and stat a new TLS session establishment and repeat test steps 2 and 3.
9. **[T]** Configure with TLS cert, Invalid MICA Policy Mapping Non-Critical, and stat a new TLS session establishment and repeat test steps 2 and 3.
10. **[T]** Configure with TLS cert, Self-signed Cert, and stat a new TLS session establishment and repeat test steps 2 and 3.

Pass/Fail Criteria

- **[T]** The testing device successfully configured itself to use certificate chain length of two (SERCA->Device Certificate).
- **[C]** The Client successfully established a TLS HTTP session using TLS cert chain length of two by conforming to the requirements specified in RFC 5246, section 7.4. Verify by inspecting the TLS packets, including verification that `TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8` cipher suite was used and cert chain length. Successfully sent a TLS based HTTP GET request to the Server `DeviceCapability` resource using the known IP address, port number, and `DeviceCapability` URI.
- **[S]** Server successfully established a TLS HTTP session using TLS cert chain length of two by conforming to the requirements specified in RFC 5246, section 7.4. Verify by inspecting the TLS packets, including verification that `TLS_ECDHE_ECDSA_WITH_AES_128_CCM_8` cipher suite was used and cert chain length. Successfully received the TLS based HTTP GET request and responded with the `DeviceCapability` resource payload as the HTTP GET response.
- **[U]** The testing device successfully configured itself to use a certificate chain length of three (SERCA->MICA->Device Certificate). The Client and Server successfully passed Criteria 2 and 3, respectively, where chain length is three, instead of two.
- **[U]** The testing device successfully configured itself to use certificate chain length of four (SERCA->MCA->MICA->Device Certificate). The Client and Server successfully passed Criteria 2 and 3, respectively, where chain length is four, instead of two.
- **[U]** The testing device successfully configured itself to use an Invalid MICA Extended Key Critical cert but failed to receive a TLS connection. The device uder test responded with a TLS Alert indicating the invalid cert and failed to establish TLS connection with the testing device.
- **[U]** The testing device successfully configured itself to use an Invalid MICA Name Non-Critical cert but failed to receive a TLS connection. The device under test responded with a TLS Alert indicating the invalid cert and failed to establish TLS connection with Client.

- **[U]** The testing device successfully configured itself to use an Invalid MICA Policy Mapping Non-Critical cert but failed to receive a TLS connection. The device under test responded with a TLS Alert indicating the invalid cert and failed to establish TLS connection with the testing device.
- **[U]** The testing device successfully configured itself to use a self-signed cert but failed to receive a TLS connection. The device under test responded with a TLS Alert indicating the invalid cert and failed to establish TLS connection with testing device.

6 Core Function Set Tests

This section specifies the tests for the CSIP IEEE 2030.5 Implementation Details section. These tests include:

- WADL
- Subscription
- Polling
- Time
- Device capability
- End device
- Function Set Assignments (FSA)
- Distributed Energy Resource (DER)
- DER Controls (DERC)
- DER Settings (DERSettings)
- Metering
- Randomization
- Responses

These tests ensure the client and server implementations conform to core function set requirements and are a prerequisite for inverter/CSIP-related tests specified in subsequent sections of this document.

CORE-001 - HTTP Request [S]

Purpose

Verify server responds correctly for all request types for the configured resource tree.

The HTTP header test verifies required HTTP header and method handling for designated IEEE 2030.5 server resources.

Setup

1. **[S]** Find which functions are supported by the Server and create the resources on the Server.
2. **[S]** Make sure the `DeviceCapability` resource includes the resources created in Step 1.
3. **[T]** Record the Client/Server communications.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[TC]** Retrieve the `DeviceCapability` resource from the Server using the supported HTTP and IP address.
3. **[TC]** Retrieve the `DeviceCapability` resource from the Server using the supported HTTPS and IP address.
4. **[TC]** Process the retrieved `DeviceCapability` resource and select an item in the `DeviceCapability` resource. For example, `DERProgram`.
5. **[TC]** Based on the WADL definition for the resource as defined in the IEEE 2030.5 standard, do GET, PUT, POST, DELETE operation as permitted using correct HTTP headers and payload (if required).
6. **[TC]** Based on the WADL definition for the resource as defined in the IEEE 2030.5 standard, do GET, PUT, POST, DELETE operation as disallowed using correct HTTP headers and payload (if required). The Server shall correctly return a `405 Method Not allowed` HTTP response for the invalid operation.
7. **[TC]** Based on the WADL definition for the resource as defined in the IEEE 2030.5 standard, do GET operation as permitted with incorrect HTTP header. The Server shall correctly return a `400 Bad Request` HTTP response for the invalid operation.
8. **[TC]** Do an invalid method operation, such as HTTP FOO, on the same resource. The Server shall correctly return a `501 Not Implemented` HTTP response for the invalid method used.
9. Repeat this test by iterating through all subordinate resources from the parent resource presented in the `DeviceCapability` payload.

Pass/Fail Criteria

- **[S]** Client requested and received the `DeviceCapability` resource on the Server using the HTTP configuration information provided. Server responded with `200 OK` and returned a conformant payload for its `DeviceCapability`.
- **[S]** Client requested and received the `DeviceCapability` resource on the Server using the HTTPS configuration information provided. The Server successfully returns a XML (or EXI) payload for its `DeviceCapability`, which is XML Schema valid and its HTTP header includes the correct values, including `200` response code, `Content-Type`, and `Content-Length`.
- **[S]** The Client successfully processed the `DeviceCapability` payload and selected the third resource from the payload to do subsequent HTTP operation. The Client processed all related attributes associated with the resource, such as `href` and `all` elements, to issue the correct subsequent HTTP operation.
- **[S]** The Client successfully did the selected HTTP operation on the resource and received the resource payload from the Server. The Client then processed the received payload and traversed dependent resources to the top resource.
- **[S]** The Client successfully requested the selected invalid HTTP operation and received the correct `405 Method Not Allowed` response. The Client processed this error response and stops issuing the same invalid request.
- **[S]** The Client successfully requested the valid HTTP operation with invalid HTTP header and received the correct `400 Bad Request` response. The Client processed this error response and stops issuing the same invalid request.
- **[S]** The Client successfully requested the invalid HTTP method and received the correct `501 Not Implemented` response. The Client processed this error response and stops issuing the same invalid request.
- **[S]** Client selected the next resource presented in the original `DeviceCapability` response and repeated the same sequence of tests.

CORE-002 - HTTP Response [S]

Purpose

Verify server responds correctly to valid and invalid requests. Verify client can follow a resource redirection.

The HTTP response test verifies required HTTP response code and method handling for designated IEEE 2030.5 server resources.

Setup

1. **[S]** Find which functions are supported by the Server and create the resources on the Server. There should be at least one resource that supports the `301 Moved Permanently` response.
2. **[S]** Make sure the `DeviceCapability` resource includes the resources created in Step 1.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[TC]** Retrieve the `DeviceCapability` resource from the Server using the supported HTTP and IP address, which shall return a `200 OK` response from the server.
3. **[TC]** Select a resource that supports a POST or PUT. Next, perform an HTTP POST or PUT operation on that resource which should trigger a `201 Created` or `204 No Content` response.
4. **[TC]** Select a resource that supports a POST or PUT. Next, perform an HTTP GET operation on that resource with missing Host header in the HTTP header, which should trigger a `400 Bad Request` response.
5. **[TC]** Select any resource that supports GET. Next, perform an HTTP GET operation on that resource with invalid URI, which should trigger a `404 Not Found` response.
6. **[TC]** Select any resource that does not support POST. Next, perform an HTTP POST operation on that resource which should trigger a `405 Method Not Allowed` response with Allow header with a list of valid methods for that requested resource.
7. **[TC]** Select any resource that supports GET. Next, perform an HTTP FOO on that resource which should trigger a `501 Not Implemented` response.
8. Repeat Steps 2 through 7 on resources subordinate to the parent resource included in the `DeviceCapability` payload.

Pass/Fail Criteria

- **[S]** Client requested and received the `DeviceCapability` resource on the Server using the HTTP configuration information provided. Server responded with `200 OK` and returned a conformant payload for its `DeviceCapability`.
- **[S]** The Client successfully requested the PUT or POST operation and received the correct `201 Created` or `204 No Content` response. The Server correctly processed

the new resource PUT or POST, and updated all relevant resources associated with new resource.

- **[C]** The Client successfully requested a GET operation on a resource that has moved. The Server successfully sent the correct `301 Moved Permanently` response, which included the new Location header. The Client successfully received the response, processed the new Location header, and performs a new PUT or POST to that new location.
- **[S]** The Client successfully requested the valid HTTP operation with invalid HTTP header and received the correct `400 Bad Request` response. The Client processed **[S]** this error response and stops issuing the same invalid request.
- **[S]** The Client successfully requested the valid HTTP operation with invalid HTTP header and received the correct `400 Bad Request` response. The Client processed this error response and stops issuing the same invalid request.
- **[S]** The Client successfully requested the selected invalid HTTP operation and received the correct `405 Method Not Allowed` response. The Client processed this error **[S]** response and stops issuing the same invalid request.
- **[S]** The Client successfully requested the invalid HTTP method and received the correct `500 Not Implemented` response. The Client processed this error response and stops issuing the same invalid request.
- **[S]** Client selected the subordinate resources to the parent resource and repeated the same sequence of tests.

CORE-003 - Polling Interaction [C, A]

Purpose

Verify client observes polling rates specified in polled resources.

The polling interaction test verifies event polling for DER resources according to CSIP polling requirements. Polling is a mandatory method for getting information from a server, as specified in the IEEE 2030.5 standard. A subscription/notification mechanism is recommended by CSIP to optimize network traffic.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Pre-register an `EndDevice` instance for the Client device, including all its required attributes. For example, `SFDI, LFDI,` and `PIN` stored in `RegistrationLink`.
3. **[S]** Create a group assignment for the preregistered `EndDevice` following a multiple layer topology-based grouping as stated in the CSIP guide.
4. **[S]** Specify the `pollRate` for each of the IEEE 2030.5 resources in a way that will determine if the Client is observing the specified poll rate for each resource

Procedure

- **[T]** Record the Client/Server communications.
- **[C]** Retrieve the `DeviceCapability` resource from the Server using the supported HTTP and IP address and find the `EndDeviceListLink` element.
- **[C]** Perform an HTTP GET operation on the `EndDeviceListLink` URI and search through the `EndDeviceList` payload to find if an `EndDevice` instance is included that matches the identity of the Client device. For example, `SFDI/LFDI`.
- **[C]** Process the `EndDevice` instance returned by the Server and perform an HTTP GET operation on the `RegistrationLink` to retrieve the `Registration:PIN` assigned to the Client. Confirm the PIN is `111115`.
- **[C]** Process the `EndDevice` instance returned by the Server and perform an HTTP GET operation on the `FunctionSetAssignmentsListLink` to retrieve the various FSA assigned resources assigned to the Client.
- **[C]** Using the `FunctionSetAssignments` instance, perform an HTTP GET operation on the `DERProgramListLink` to retrieve the `DERProgramList` assigned to the Client.
- **[C]** Parse the returned `DERProgramList` payload to find how many `DERProgram` instances are included.
- **[C]** Set the `pollRate` attribute in each of `DeviceCapability`, `EndDeviceList`, `FunctionSetAssignmentsList`, `DERProgramList` to verify the ability to observe the specified poll rate.

Pass/Fail Criteria

- **[C]** Client requested and received the `DeviceCapability` resource on the Server using the HTTP configuration information provided. Server responded with `200 OK` and returned a conformant payload for its `DeviceCapability`.
- **[C]** Client did a successful HTTP GET on the URI of its `EndDevice` instance, searched through the instance, and found the `RegistrationLink`. The Server successfully received the HTTP GET request on the `EndDevice` ref from the Client and returned a valid resource payload (includes `RegistrationLink` with PIN attribute) as a response to the HTTP GET request from the Client.
- **[C]** The Client successfully processed the received Registration resource from the Server, found the PIN attribute in the resource, and verified it is same as what its own PIN number is.
- **[C]** The Client successfully issued an HTTP GET on the `FunctionSetAssignmentsListLink` and received a conformant `FunctionSetAssignmentsList` payload from the Server. The Server successfully received the HTTP GET request on `FunctionSetAssignmentsListLink` and returned a conformant payload as a response.
- **[C]** The Client successfully did an HTTP GET on the `DERProgramListLink` from the `FunctionSetAssignments` instance and received a conformant `DERProgramList` assigned to the Client. The Server successfully received the HTTP GET request on the `DERProgramListLink` and returned a conformant payload as a response.
- **[C]** The Client successfully iterated through all `DERProgram` instances and applied the `DERProgram` list priority processing to select the highest priority `DERProgram` to issue the next HTTP GET request to.
- **[C]** The Client successfully did an HTTP GET on the `DERProgram` selected and process the returned payload. Processed the elements in the `DERProgram` payload and did HTTP GET operations to retrieve subordinate resources. If there is an active DER Event applicable to the Client, the Client scheduled the event internally in its DER system and participated based on the provided schedule.
- **[C]** Client polled each of the specified resources with a `pollRate` attribute at the specified poll rate.

CORE-004 - List Handling [S]

Purpose

Verify server can handle each of the query parameters associated with list handling.

The list handling test verifies list query and response handling. List and query string search features are an essential part of IEEE 2030.5 list-based resources, which include many DER-related resources.

Setup

1. **[S]** Create an `EndDeviceList` with at least three `EndDevice` instances in the list to exercise the query string parameter searching operation. If the Server cannot meet this condition, create `FunctionSetAssignmentsList` or `DERProgramList` with required subordinate resources.
2. **[S]** Make sure the `DeviceCapability` resource has the link for `EndDeviceList` (or `FunctionSetAssignmentsList` or `DERProgramList`) for its clients.

Procedure

Query string parameters are described in the IEEE 2030.5 Application Protocol Standard under Section 6.6 List Resources. Readers are advised to review this section in detail before running this test.

Substitute `EndDeviceList` in the HTTP GET commands below with actual URI returned in the DeviceCapability response payload for the list resource you are working with.

1. **[T]** Record the Client/Server communications.
2. **[TC]** Retrieve the `DeviceCapability` resource from the Server using the supported HTTP and IP address.
3. **[TC]** Process the retrieved `DeviceCapability` resource and perform an HTTP GET operation on the `EndDeviceList` (or `FunctionSetAssignmentsList` or `DERProgramList`) resource link included in the `DeviceCapability` with query string parameter. For example, `HTTP GET http://ipaddress/EndDeviceList?s=0`. This query string search shall return the first element in the list.
4. **[TC]** Perform an HTTP GET operation on the `EndDeviceList` (or `FunctionSetAssignmentsList` or `DERProgramList`) resource link included in the `DeviceCapability` with no query string parameter. For example, `HTTP GET http://ipaddress/EndDeviceList`. This request shall return the first element in the list.
5. **[TC]** Perform an HTTP GET operation on the `EndDeviceList` (or `FunctionSetAssignmentsList` or `DERProgramList`) resource link included in the `DeviceCapability` with time-based query string parameter. For example, `HTTP GET http://ipaddress/EndDeviceList?a=time`, where `time` is current time expressed as a valid `TimeType`. This request shall return the first element in the list (if

the list primary key is not time-based) or the element afer the specified a=time in the time keyed list (if the list primary key is time-based).

6. **[TC]** Perform an HTTP GET operation on the `EndDeviceList` (or `FunctionSetAssignmentsList` or `DERProgramList`) resource link included in the `DeviceCapability` with query string parameter. For example, `HTTP GET http://ipaddress/EndDeviceList?l=0`. This request shall return an empty list.
7. **[TC]** Perform an HTTP GET operation on the `EndDeviceList` (or `FunctionSetAssignmentsList` or `DERProgramList`) resource link included in the `DeviceCapability` with query string parameter. For example, `HTTP GET http://ipaddress/EndDeviceList?l=1000`. This request shall return upto 1000 elements in the list - depending on the number of elements that exist in the requested list at test execution time.
8. **[TC]** Perform an HTTP GET operation on the `EndDeviceList` (or `FunctionSetAssignmentsList` or `DERProgramList`) resource link included in the `DeviceCapability` with time-based query string parameter. For example, `HTTP GET http://ipaddress/EndDeviceList?s=3&a=time`, where `time` is current time expressed in `TimeType`. This request shall return the contents of the ordinal elements in the list beginning s=3 since none of these are time based.
9. **[TC]** Perform an HTTP GET operation on the `EndDeviceList` (or `FunctionSetAssignmentsList` or `DERProgramList`) resource link included in the `DeviceCapability` with query string parameter. For example, `HTTP GET http://ipaddress/EndDeviceList?s=1&l=1&s=2`. This request shall return the contents of the ordinal elements in the list beginning with the third and ignoring the second, s=2.
10. **[C]** Perform an HTTP GET operation on the `EndDeviceList` (or `FunctionSetAssignmentsList` or `DERProgramList`) resource link included in the `DeviceCapability` with query string parameter. For example, `HTTP GET http://ipaddress/EndDeviceList?l=0&l=2`. This request shall return an empty list.
11. **[TC]** Perform an HTTP GET operation on the `EndDeviceList` (or `FunctionSetAssignmentsList` or `DERProgramList`) resource link included in the `DeviceCapability` with time-based query string parameter. For example, `HTTP GET http://ipaddress/EndDeviceList?s=1&a=time&a=timeplus-two4 hours`, where `time` is current time expressed in `TimeType`. This request shall return contents of the ordinal elements in the list beginning with the s=1 query string since none of these lists are time based.
12. **[TC]** Perform an HTTP GET operation on the `EndDeviceList` (or `FunctionSetAssignmentsList` or `DERProgramList`) resource link included in the `DeviceCapability` with query string parameter. For example, `HTTP GET`

http://ipaddress/EndDeviceList?l=1&b=2. This request shall return contents of the list starting with the first element and only that element.

Pass/Fail Criteria

- **[S]** Client requested and received the DeviceCapability resource on the Server using the HTTP configuration information provided. Server responded with 200 OK and returned a conformant payload for its DeviceCapability.
- **[S]** Client processes the retrieved DeviceCapability resource and does an HTTP GET on the EndDeviceList (or FunctionSetAssignmentsList or DERProgramList) resource link included in the DeviceCapability with query string parameter. For example, HTTP GET http://ipaddress/EndDeviceList?s=0. This query string search shall return the third element in the list.
- **[S]** Client does an HTTP GET on the EndDeviceList, or DERProgramList or FunctionSetAssignmentsList, resource link included in the DeviceCapability with no query string parameter. For example, HTTP GET http://ipaddress/EndDeviceList, which shall return the third element in the list.
- **[S]** Client does an HTTP GET on the EndDeviceList (or FunctionSetAssignmentsList or DERProgramList) resource link included in the DeviceCapability with time-based query string parameter. For example, HTTP GET http://ipaddress/EndDeviceList?a=time, where time is current time expressed as a valid TimeType. This request shall return the first element in the list (if the list primary key is not time-based) or the element after the specified a=time in the time-keyed list (if the list primary key is time-based).
- **[S]** Client does an HTTP GET on the EndDeviceList, or DERProgramList or FunctionSetAssignmentsList, resource link included in DeviceCapability with a time-based query string parameter. For example, HTTP GET http://ipaddress/EndDeviceList?l=0. This request shall return an empty list.
- **[S]** Client does an HTTP GET on the EndDeviceList (or FunctionSetAssignmentsList or DERProgramList) resource link included in the DeviceCapability with time-based query string parameter. For example, HTTP GET http://ipaddress/EndDeviceList?l=1000. This request shall return up to 1000 elements in the list - depending on the number of elements that exist in the requested list at test execution time.
- **[S]** Client does an HTTP GET on the EndDeviceList (or FunctionSetAssignmentsList or DERProgramList) resource link included in the DeviceCapability with time-based query string parameter. For example, HTTP GET http://ipaddress/EndDeviceList?s=3&a=time, where time is current time expressed in TimeType. **[S]** Client does an HTTP GET on the EndDeviceList (or FunctionSetAssignmentsList or DERProgramList) resource link included in the DeviceCapability with time-based query string parameter. For example, HTTP

`GET http://ipaddress/EndDeviceList?s=1&l=1&s=2`. This request shall return the contents of the ordinal elements in the list beginning with the third and ignoring the second, s=2.

- **[S]** Client does an HTTP GET on the `EndDeviceList`, or `DERProgramList` or `FunctionSetAssignmentsList`, resource link included in the `DeviceCapability` with time-based query string parameter. For example, `HTTP GET http://ipaddress/EndDeviceList?l=0&l=2`, which shall return an empty list.
- **[S]** Client does an HTTP GET on the `EndDeviceList` (or `FunctionSetAssignmentsList` or `DERProgramList`) resource link included in the `DeviceCapability` with time-based query string parameter. For example, `HTTP GET http://ipaddress/EndDeviceList?s=1&a=time&a=timeplus-two4 hours`, where `time` is current time expressed in `TimeType`. **[S]** Client does an HTTP GET on the `EndDeviceList` (or `FunctionSetAssignmentsList` or `DERProgramList`) resource link included in the `DeviceCapability` with time-based query string parameter. For example, `HTTP GET http://ipaddress/EndDeviceList?l=1&b=2`. This request shall return contents of the list starting with the third element and only that element.

CORE-005 - Basic Time [C, A, S]

Purpose

The basic time test verifies the discovery and usage of the `Time` resource provided by a time server and using the `Time` resource to synchronize with client devices.

Setup

1. **[S]** Verify a Time resource (with quality metric=7 for Intentionally Uncoordinated time) can be created (or exists) on the Server.
2. **[S]** Make sure the `DeviceCapability` resource has the link for Time on the Server.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Retrieve the `DeviceCapability` resource from the Server using the supported HTTP and IP address.
3. **[C]** Process the retrieved `DeviceCapability` resource and verify the Time resource available from the Server and included as a link in the `DeviceCapability` resource.
4. **[S]** Confirm the quality metric of the Time resource is 7.
5. **[C]** Retrieve the Time resource from the Server by using the href information included in the `DeviceCapability` time resource.
6. **[C]** Process the retrieved Time resource, verify its quality metric to be 7 (intentionally uncoordinated) and synchronize the Client time using the information and display the updated time information on the Client display, if supported.

Pass/Fail Criteria

- **[C]** Client requested and received the `DeviceCapability` resource on the Server using the HTTP configuration information provided. Server responded with `200 OK` and returned a conformant payload for its `DeviceCapability`.
- **[S]** Server included the Time resource link in the returned `DeviceCapability` to the Client. Client finds the href information for the Time resource.
- **[C]** The Client successfully retrieves the Time resource using the href information and receives a successful `200 OK` response.
- **[C]** The Client successfully parses the received Time resource, verifies the quality metric to be `7`. The Client successfully processes the contents of the Time resource and synchronizes its time using the content information. The Client displays the updated Time information to its display, if such display is supported.

CORE-006 - Advanced Time [S]

Purpose

Verify time change on server and that the time change is acquired. Verify the time change is logged.

The advanced time test verifies that the Client can obtain the time from the Server and handle advanced scenarios, such as not honoring rolled back clock times.

Setup

1. **[S]** Verify that a Time resource can be created, or exists, on the Server.
2. **[S]** Make sure the `DeviceCapability` resource has the link for Time on the Server.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Retrieve the `DeviceCapability` resource from the Server using the supported HTTP and IP address.
3. **[C]** Process the retrieved `DeviceCapability` resource and verify the Time resource available from the Server and included as a link in the `DeviceCapability` resource.
4. **[C]** Retrieve the Time resource from the Server by using the href information included in the `DeviceCapability` Time resource.
5. **[S]** Change the time forward by 1 hour which should trigger a `LogEvent` indicating `TM_TIME_ADJUSTED`.
6. **[C]** Retrieve the Time resource from the Server by using the href information included in the `DeviceCapability` Time resource to check the updated time.
7. **[C]** Retrieve the `LogEventList` from the Server `SelfDevice` and verify a `TM_TIME_ADJUSTED` `LogEvent` was generated.

Pass/Fail Criteria

- **[C]** Client requested and received the `DeviceCapability` resource on the Server using the HTTP configuration information provided. Server responded with `200 OK` and returned a conformant payload for its `DeviceCapability`.
- **[S]** Server included the Time resource link in the returned `DeviceCapability` to the Client. Client finds the href information for the Time resource.
- **[C]** The Client successfully retrieves the Time resource using the href information and receives a successful `200 OK` response.
- **[S]** Server detects the one hour forward time changed generates a `LogEvent` indicating `TM_TIME_ADJUSTED` value.
- **[C]** The Client successfully retrieves the Time resource using the href information and receives a successful `200 OK` response and validates the updated time (one hour forward time).

CORE-009 - Advanced End Device [C, A, S]

Purpose

Verify end device resources are provided and acquired including DERCapability, DERSettings, DERStatus, and DERAvailability. Verify client updates DERCapability, DERSettings, DERStatus, and DERAvailability.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Pre-register an `EndDevice` instance for the Client device, including all its required attributes. For example, `SFDI/LFDI` and PIN which is stored in the `RegistrationLink`.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Retrieve the `DeviceCapability` resource from the Server using the supported HTTP and IP address and find the `EndDeviceListLink` element.
3. **[C]** Perform an HTTP GET operation on the `EndDeviceListLink` URI and search through the `EndDeviceList` payload to find if an `EndDevice` instance is included that matches the identity of the Client device. For example, `SFDI/LFDI`. If found, skip next step.
4. **[C]** Using the Location of the created `EndDevice` instance returned by the Server, perform an HTTP GET operation on that Location. On successful GET operation, perform an HTTP GET operation on the `RegistrationLink` href to find the PIN value for the Client device.
5. **[C]** Process the returned Registration resource and search for the PIN element and find its value. Verify the PIN value is the same PIN value the Client device has preregistered.
6. **[C]** Using the `EndDevice` instance resource returned in step 4, find the `DERListLink` information. The `DERListLink` resource has information about the DER device state, capabilities and settings. Do an HTTP PUT on the `DERListLink` using the href attribute and updated values for `DERCapabilities`, `DERSettings`, `DERStatus` or `DERAvailability`.

Pass/Fail Criteria

- **[C]** Client requested and received the `DeviceCapability` resource on the Server using the HTTP configuration information provided. Server responded with `200 OK` and returned a conformant payload for its `DeviceCapability`.
- **[S]** Server returned an `EndDeviceList` payload in response to the Client HTTP GET request. If the Client is preregistered, the Server should include an `EndDevice` instance associated with it. Otherwise, the `EndDeviceList` payload will not include the

instance for the Client in which case the Client shall POST its own instance to the Server.

- **[C]** Client did a successful HTTP GET on the URI of its `EndDevice` instance, searched through the instance, and found the `RegistrationLink`. The Server successfully received the HTTP GET request on the `EndDevice` href from the Client and returned a valid resource payload (includes `RegistrationLink` with PIN attribute) as a response to the HTTP GET request from the Client.
- **[C]** The Client successfully processed the received Registration resource from the Server, found the PIN attribute in the resource, and verified it is same as what its own PIN number is.
- **[C]** The Client successfully searched its `EndDevice` instance resource and found the `DERListLink` resource which can be used to PUT updated information to. Successfully packaged the updated `DERCapabilities`, `DERSettings`, `DERStatus` or `DERAvailability` resource payload and did an HTTP PUT to the Server using the updated resources. The Server successfully received the updated resource from the HTTP PUT operation and updated its internal resources to reflect the updated information.
- **[C]** The Client successfully did an HTTP GET on the `DERListLink`-related update it did in the previous step and received a `200 OK` with the response payload from the Server. The Server successfully received an HTTP GET request from the Client and responded with an HTTP `200 OK` response and relevant payload.

CORE-010 - Function Set Assignments [C, A, S]

Purpose

The function set assignments test verifies that the Client can get its `FunctionSetAssignments` from the Server. CSIP requires the Client to support a seven-level deep group hierarchy as shown in Figure 1.

Each group has an associated `DERProgram`, and each end device is assigned a `DERProgramList` with all the `DERPrograms`, or groups, to which it belongs. The `DERProgramList` link for each end device is provided using each end device `FunctionSetAssignment`.

This test and all DER event tests configure the end device Client with an `EndDevice` instance that points to a `FunctionSetAssignmentsList` for that end device. This `FunctionSetAssignmentsList` includes a `FunctionSetAssignment` instance. The `FunctionSetAssignment` instance includes a `DERProgramList` link with all the `DERPrograms`, groups, to which the end device belongs.

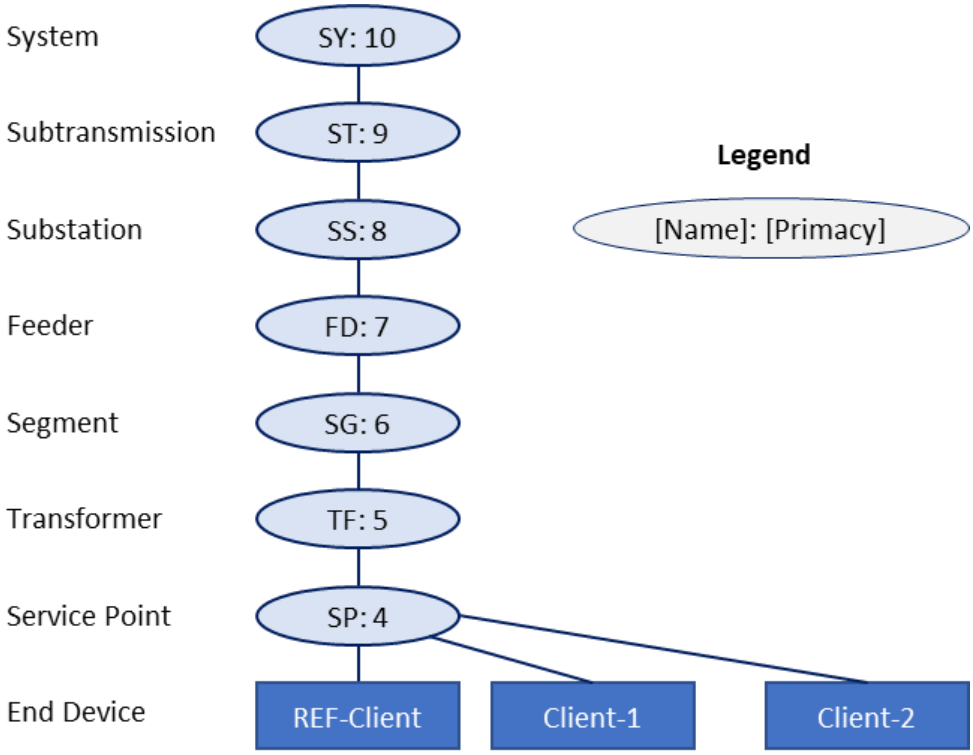

Figure 1 Function Set Assignments

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client. Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
   - `SFDI`
     Short form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the EUT.
   - `LFDI`
     Long form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `LFDI` of the EUT.
   - `RegistrationLink`
     Link to the Registration resource of the `EndDevice` instance under test with the `Registration:PIN` resource set to `111115`.
   - `FunctionSetAssignmentsListLink`
     Link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
4. **[S]** Construct the groups, such as `DERPrograms`, with the associated primacy values as shown in Figure 1.
5. **[S]** Construct the `DERProgramList` for the Client.
6. **[S]** Construct a `FunctionSetAssignments` resource for the Client, which has the `DERProgramListLink` and `TimeLink`.
7. **[S]** Construct a `FunctionSetAssignmentsList` with the `FunctionSetAssignments` instance.
8. **[S]** For the Client `EndDevice` instance, assign the `FunctionSetAssignmentsListLink` to the `FunctionSetAssignmentsList`.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[S]** Configure resources as outlined in Setup and wait for Client to acquire resources.
3. Wait two minutes after resources before terminating the test.
4. **[C]** Retrieve the `DeviceCapability` resource from the Server using the supported HTTP and IP address and find the `EndDeviceListLink` element.
5. **[C]** Perform an HTTP GET operation on the `EndDeviceListLink` URI and search through the `EndDeviceList` payload to find if an `EndDevice` instance is included that matches the identity of the Client device. For example, `SFDI/LFDI`.

6. **[C]** On successful identity match, perform an HTTP GET operation on the `RegistrationLink` href to find the PIN value for the Client device.
7. **[C]** Process the returned Registration resource and search for the PIN element and find its value. Verify the PIN value is the same PIN value the Client device has preregistered.
8. **[C]** Using the `EndDevice` instance resource returned in step 4, find the `FunctionSetAssignmentsListLink` information.
9. **[C]** Starting with the `FunctionSetAssignmentsListLink`, follow the link to find (1) the `FunctionSetAssignmentsList`, (2) the `FunctionSetAssignments` instance, (3) the `DERProgramListLink`, (4) the `DERProgramList`, and (5) the `DERPrograms`.
10. **[C]** GETs all the `DERPrograms`, or groups, assigned to it.

Pass/Fail Criteria

- **[C]** The Client successfully searched its `EndDevice` instance resource and found its `FunctionSetAssignmentsListLink` resource
- **[C]** The Client successfully did an HTTP GET on the `FunctionSetAssignmentsList`, the `FunctionSetAssignments` instance, the `DERProgramList`, and all seven `DERPrograms`.
- **[S]** For all the queries, the Client received a `200 OK` with the response payload from the Server.

CORE-011 - Advanced Function Set Assignments [C, A, S]

Purpose

Verify provisioning and acquisition of end device with a fifteen-level resource hierarchy.

The advanced function set assignments test verifies that the Client can get its `FunctionSetAssignments` from the Server.

CSIP requires the Client to support up to a fifteen deep hierarchy so the Client must be able to support fifteen FSAs.

Setup

1. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client. Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
2. **[S]** For each `EndDevice`, these fields must be populated:
    - `LFDI`
      Long form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the Client.
    - `SFDI`
      Short form device identifier. One of the three EndDevice instances must have an `SFDI` that matches the `LFDI` of the Client.
    - `RegistrationLink`
      Link to the Registration resource of the `EndDevice` instance under test with the `Registration:PIN` resource set to `111115`.
    - `FunctionSetAssignmentsListLink`
      Link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
3. **[S]** Construct a `FunctionSetAssignmentsList` resource for the Client with fifteen `FunctionSetAssignments` following the CSIP topology/non-topology recommendations.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Retrieve the `DeviceCapability` resource from the Server using the supported HTTP and IP address and find the `EndDeviceListLink`element.
3. **[C]** Perform an HTTP GET operation on the `EndDeviceListLink` URI and search through the `EndDeviceList` payload to find if an `EndDevice` instance is included that matches the identity of the Client device. For example, `SFDI/LFDI`. If found, skip next step.
4. **[C]** Using the Location of the created `EndDevice` instance returned by the Server, perform an HTTP GET operation on that Location. On successful GET operation, the `EndDevice` instance payload returned by the Server shall include relevant subordinate resources assigned to the Client. For example, `SubscriptionListLink`, `RegistrationLink`, and `FunctionSetAssignmentsListLink`, `SFDI/LFDI`, and others.
5. **[C]** Process the `EndDevice` instance returned by the Server and perform an HTTP GET operation on the `RegistrationLink` to retrieve the `Registration:PIN` assigned to the Client. Confirm the PIN is `111115`.
6. **[C]** Process the `EndDevice` instance returned by the Server and perform an HTTP GET operation on the `FunctionSetAssignmentsListLink` to retrieve the various FSA assigned resources assigned to the Client.
7. **[C]** If there is more than one `FunctionSetAssignments` in the `FunctionSetAssignementsList` payload, do additional HTTP GET on the rest of the `FunctionSetAssignments` and sort it based on the Primary Key (`mRID`) to find the priority ordering.
8. **[C]** Using the highest priority `FunctionSetAssignments` instance, perform an HTTP GET operation on the `DERProgramListLink` to retrieve the various `DERProgramList` assigned to the Client. If no `DERProgramListLink` is found, go back to step 7 and select the next higher priority `FunctionSetAssignments` instance.
9. **[C]** Parse the returned `DERProgramList` payload to find how many `DERProgram` instances are included. If there is more than one, iterate through all `DERProgram` instances and apply the `DERProgram` list priority processing by examining the Primary key (primacy) and Secondary key (`mRID`) to select the highest priority `DERProgram`.
10. **[C]** Perform an HTTP GET operation on the `DERProgram` selected and process the returned payload. Process the elements in the `DERProgram` payload and do HTTP GET operations to get subordinate resources. When the DER event is found through `DERControl`, schedule it on the Client based on the `DERControl` attributes.

Pass/Fail Criteria

- **[C, S]** Client requested and received the `DeviceCapability` resource on the Server using the HTTP configuration information provided. Server responded with `200 OK` and returned a conformant payload for its `DeviceCapability`.
- **[C, S]** Server returned an `EndDeviceList` payload in response to the Client HTTP GET request.
- **[C, S]** Client did a successful HTTP GET on the URI of its `EndDevice` instance and searched through the instance and found the `RegistrationLink`. The Server successfully received the HTTP GET request on the `EndDevice` href from the Client and returned a valid resource payload (includes `RegistrationLink` with PIN attribute) as a response to the HTTP GET request from the Client.
- **[C, S]** The Client successfully processed the received Registration resource from the Server and found the PIN attribute in the resource and verified it is same as what its own PIN number is.
- **[C, S]** The Client successfully issued an HTTP GET on the `FunctionSetAssignmentsListLink` and received a conformant `FunctionSetAssignmentsList` payload from the Server. The Server successfully received the HTTP GET request on `FunctionSetAssignmentsListLink` and returned a conformant payload as a response.
- **[C, S]** The Client successfully iterated through all `FunctionSetAssignments` instances, sorted them based on Primary/Secondary key and selected the highest priority `FunctionSetAssignments` instance.
- **[C, S]** The Client successfully did an HTTP GET on the `DERProgramListLink` from the selected `FunctionSetAssignments` instance and received a conformant `DERProgramList` assigned to the Client. The Server successfully received the HTTP GET request on the `DERProgramListLink` and returned a conformant payload as a response.
- **[C, S]** The Client successfully iterated through all `DERProgram` instances and applied the `DERProgram` list priority processing to select the highest priority `DERProgram` for the next HTTP GET request.
- **[C, S]** The Client successfully did an HTTP GET on the `DERProgram` selected and process the returned payload. Processed the elements in the `DERProgram` payload and did HTTP GET operations to retrieve subordinate resources. If there is an active DER Event applicable to the Client, the Client scheduled the event internally in its DER system and participated based on the provided schedule.

CORE-012 - Basic DER Program/Control [C, A, S]

Purpose

The basic DER program/control test verifies that the Client can process the DERProgram provided by the IEEE 2030.5 server through FSA/EndDevice allocation. This test shall include list processing, list storage, and inverter control modes.

Setup

1. **[S]** Setup following the CORE-010 - Function Set Assignments test, which should setup the Server with EndDevice and FSAs for the Client device.
2. **[S]** Construct one DERProgramList, which includes one DERPrograms and one DERControl (and required subordinate resources):
    - DERProgram1/DERControl1 has at minimum of one immediate control mode and one curve-based control mode. It shall start three minutes from now with duration of two minutes and have highest Primacy value (lowest priority).
    - At least one DERProgram shall have DefaultDERControl attribute and other requirements listed above.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Process the received DERProgramList assigned to the Client through the FSA instance and find the highest priority DERProgram /DERControl to process by following the list ordering rules. The DER-Client shall schedule the selected DERProgram/DERControl (should be DERProgram 1, above).
3. **[C]** After the correct DERProgram/DERControl is selected, Client shall issue subsequent HTTP GET requests to retrieve the subordinate resources associated with the DERProgram and DERControl, such as DERCurveListLink with one or more DERCurveList instances.
4. **[C]** For the DERCurve resources in the DERCurveListLink, the Client shall do HTTP GET requests to retrieve the 10 curve points and their values. The Client shall use these curve point and related values to construct the DER curve-based control, which shall be communicated to the internal inverter system and the schedule information. For example, start time and duration.
5. **[C]** Until the selected DERProgram/DERControl becomes active, the Client shall use the DefaultDERControl value by setting the DER control mode using the DefaultDERControl attribute.
6. **[C]** Periodically poll the associated DERControl resource to check the Status of the DER event. Client completes the DER event either based on the DERControl status value and/or start time plus duration interval.

Pass/Fail Criteria

- **[C, S]** The Client successfully received a conformant, or any, `DERProgramList` payload from the Server. Client was able to find which multiple `DERPrograms` take priority by using the list ordering rules. Client was able to schedule a DER event based on the `DERProgram/DERControl` information received.
- **[C, S]** The Client successfully retrieved the subordinate resources for the `DERProgram/DERControl` payload. For example, `DERCurveList` information. The Server sent all requested resource information as a response to the Client HTTP GET requests.
- **[C, S]** The Client successfully retrieved all curve points and their values from the Server. The Server sent all curve points and their values as requested by the Client. The Client successfully constructed the curve points and their values to send the information to the DER inverter system.
- **[C, S]** The Client successfully found and scheduled the `DefaultDERControl` while no DER event was active.
- **[C, S]** The Client successfully retrieved the `DERControl` status to find the completion of the DER event and successfully completed the DER Event.

CORE-013 - Advanced DER Program/Control [C, A, S]

Purpose

The advanced DER program/control test verifies that the Client can process the DERProgram provided by the IEEE 2030.5 server through FSA/EndDevice allocation. This test shall include advanced test scenarios, such as a mix of inverter control modes (immediate/curve-based), and at least ten different DERPrograms and ten different DERControl instances.

Setup

1. **[S]** Construct 1 DERProgramList, which includes seven DERPrograms/DERControl (and required subordinate resources):
    - DERProgram#N/DERControl#N has primacy value of N, start time of plus-N minutes with duration of 30 seconds (no randomization). Use the opModFixedPFInjectW function for these DERControls.
    - At least one DERProgram shall have DefaultDERControl attribute and other requirements listed above.
    - The immediate and curve-based controls shall conform to the requirements listed above.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Process the received DERProgramList assigned to the Client through the FSA instance and find the highest priority DERProgram /DERControl to process by following the list ordering rules. The DER-Client shall schedule the selected DERProgram/DERControl (should be DERProgram 1, above).
3. **[C]** Client shall issue subsequent HTTP GET requests to retrieve the subordinate resources associated with each of the DERProgram and DERControl.
4. **[C]** Until the selected DERProgram/DERControl becomes active, the Client shall use the DefaultDERControl value by setting the DER control mode using the DefaultDERControl attribute.
5. **[C]** Repeat steps 4 & 5 for the remaining DERProgram/DERControl instances assigned to the Client device.

Pass/Fail Criteria

- **[C, S]** The Client successfully received a conformant, or any, DERProgramList payload from the Server. Client was able to find which multiple DERPrograms take priority by using the list ordering rules. Client was able to schedule a DER event based on the DERProgram/DERControl information received.
- **[C, S]** The Client successfully retrieved the subordinate resources for the DERProgram/DERControl payload. For example, DERCurveList information. The

Server sent all requested resource information as a response to the Client HTTP GET requests.

- **[C, S]** The Client successfully found and scheduled the `DefaultDERControl` while no DER event was active.
- **[C, S]** The Client successfully retrieved the `DERControl` status to find the completion of the DER event and successfully completed for each of the DER Event.

CORE-014 - Basic DER Settings (Power Generating) [C, A, S]

Purpose

The basic DER settings, power generating test verifies that the Client can manage its DER-related `DERCapability` (nameplate ratings), `DERSettings`, `DERAvailability`, and `DERStatus` with the IEEE 2030.5 server. This test shall include basic test scenarios where the client device accurately reports its set of information through the `DERInfo`/`EndDevice` attributes.

This test scenario is valid for only the direct connection method as specified by CSIP.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Pre-register an `EndDevice` instance for the Client device, including all following attributes. For example, `SFDI/LFDI`, `FunctionSetAssignments`, `DERList`, `DERCapability`, `DERStatus`, `DERAvailability`, `DERSettings`. The Server shall permit PUT to these resources based on the WADL definition.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Perform an HTTP GET operation on the `DERList` resource assigned through its pre-registered `EndDevice` instance on the Server.
3. **[C]** Do an HTTP PUT to the DER resources with the list of links (and current information populated for elements described in steps 5 through 9), which it maintains on the `DERList` resource presented through its `EndDevice` instance, such as `DERCapability`, and `DERSettings`, conformant to the above requirements.
4. **[C]** Perform an HTTP GET operation on the `DERCapability` payload for this `EndDevice` to check the `type` element to find if the device has support for power generation (see steps 5 through 9). For example, PV or combined PV/Storage. If not, skip to Reactive Power Support test later in this test.
5. **[C]** Perform an HTTP GET operation on the `DERSettings` payload for this `EndDevice`. This `DERSettings` should reflect the current settings of the individual DER Client device (`DERSettings` should have been POST/PUT from step 3 to reflect is current state).
6. **[C]** Based on the `DERCapability` and `DERSettings` payloads, find presence of mandatory items for power generation DER devices:
    - `DERCapability`/`modesSupported` must include support for `opModMaxLimW` (`modesSupported=20`) and `DERCapability`/`rtgMaxW`.
    - If power output can be adjusted, de-rated, `DERSettings` must include support for `setMaxW`.

- If apparent power output can be adjusted, de-rated, DERCapability must include rtgMaxVA and DERSettings must include setMaxVA.
- DERCapability/modesSupported must include support for opModVoltWatt curve-based controls.
- DERCapability/modesSupported should also indicate support for opModFixedVAr or opModFixedPFInjectW.
- Confirm the supported nameplate ratings the equivalent DERSettings value are less than or equal to nameplate values.

7. **[C]** Reactive Power (VAr) Test: If the DERCapability/modesSupported indicates support for opModFixedVAr, do the next test step. Otherwise, skip to Power Factor test section later in this test.
8. **[C]** Based on the DERCapability and DERSettings payloads, find presence of mandatory items for Reactive Power capable DER devices:
    - DERCapability/modesSupported must include support for opModFixedVAr.
    - DERCapability must include rtgMaxVAr and DERSettings must include setMaxVAr if the maximum reactive power output is adjustable.
    - Confirm the equivalent DERSettings values are less than or equal to nameplate values. For example, setMaxVAr <= rtgVAr.
    - DERCapability/modesSupported should include support for opModVoltVAr and rtgMaxVArNeg/setMaxVArNeg should be supported if the received VAr is significantly different from the delivered VAr.
    - Confirm the equivalent DERSettings values are less than or equal to nameplate values. For example, setMaxVArNeg >= rtgVArNeg.
    - Confirm the opModFixedVAr setpoint refType is one of the following:
        - %setMaxW
        - %setMaxVAr
        - %statVArAvail
9. **[C]** Power Factor (PF) Test: If the DERCapability/modesSupported indicates support for opModFixedPFInjectW, do the next test step. Otherwise, there are no further test steps to execute.
10. **[C]** Based on the DERCapability and DERSettings payloads, find presence of mandatory items for Power Factor capable DER devices:
    - DERCapability/modesSupported must include support for opModFixedPFInjectW.
    - DERCapability must include rtgMinPFOverExcited and DERSettings must include setMinPFOverExcited, if the minimum are adjustable.
    - Confirm the equivalent DERSettings values are less than or equal to nameplate values. For example, rtgMinPFOverExcited <= setMinPFOverExcited (positive value).

- DERCapability/modesSupported should include support for opModWattPF and rtgMinPFUnderExcited/setMinPFUnderExcited should be supported if the lagging PowerFactor is different significantly from the leading PowerFactor value.
- Confirm the equivalent DERSettings values are less than or equal to nameplate values. For example, setMinPFUnderExcited (negative) >= rtgMinPFUnderExcited.

Pass/Fail Criteria

- **[C]** Client successfully gets DERInfo and subordinate resources from the Server.
- **[C]** Client successfully stores its current DERInfo and subordinate resources with updated information to Server. The Server successfully updates its local resources to reflect the POST/PUT information from the Client.
- **[C]** Client successfully retrieves and validates the DERCapability/type attribute and confirms the Client is a Power Generation DER device.
- **[C]** Client successfully retrieves and validates the DERSettings payload to do the mandatory checks in the remaining test steps.
- **[C]** Test operator successfully confirms the various DERCapability/DERSettings validation as outlined above.
- **[C]** Reactive Power (VAr) Test: Test operator successfully validates the various DERCapability/DERSettings as outlined above.
- **[C]** Test operator successfully confirms the various DERCapability/DERSettings validation as outlined above.
- **[C]** Power Factor (PF) Test: Test operator successfully validates the various DERCapability/DERSettings as outlined above.
- **[C]** Test operator successfully confirms the various DERCapability/DERSettings validation as outlined above.

CORE-018 - Basic Subscription [A, S]

Purpose

Verify provisioning, acquisition, and execution of subscription resources.

The basic subscription test verifies:

- The IEEE 2030.5 server supports a set of resources which can be subscribed to.
- The client devices can receive and act on notifications from those resources.

This test shall include basic test scenarios, where:

- A set of resources are made available for subscribing by an IEEE 2030.5 server.
- Periodic notifications are made on a subset of those resources to a client device.

Setup

1. **[S]** The `EndDevice` resource with the `SFDI/LFDI` value of the Client device shall have a `SubscriptionListLink` element included with the appropriate attributes so the Server and Client can use the `SubscriptionListLink` to support subscription and notification requests. This link shall be empty at this stage of this test. For example, all=0.
2. **[S]** The `FSAList` belonging to the `EndDevice` instance with the Client `SFDI/LFDI` shall be able to be subscribed to with a value of one, unconditional, subscription.
3. **[S]** The server shall be able to update an attribute included in the Client `FSAList` to trigger a notification message to the Client.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Search the retrieved list of `EndDevice` instances from the Server for the Client `SFDI/LFDI` value. Next, find the `SubscriptionListLink` elements and its attribute to confirm the Server supports subscription.
3. **[C]** Search the `FSAList` assigned to the Client `EndDevice` and verify it can be subscribed to.
4. **[C]** Prepare a subscription request (with all required elements/attributes) that includes the `FSAList` and send an HTTP POST request.
5. **[S]** For the received subscription request from the Client, process and send an HTTP `201 Created` or HTTP `204 No Content` response (after validating such request can be made by the Client) with the correct HTTP Location header which indicates the URI of the created resource for the subscription request stored on the Server.
6. **[S]** Cause a change for one the elements of the FSA resource for the Client, which shall cause a notification message. At such resource change, send a notification message using the URI information sent by the Client from the previous test step and include the updated FSA resource payload.
7. **[C]** When the Notification message is received from the Server, process the incoming Notification message and payload, including validation.
8. **[C]** Perform an HTTP GET request on the href of the resource included in the Server Notification message. Process the payload returned from the Server from the HTTP GET request and compare to the Notification body payload, where they shall be identical.

Pass/Fail Criteria

- **[C, S]** The Client successfully searched the retrieved list of `EndDevice` instances from the Server and found the correct instance, which includes the Client `SFDI/LFDI` value. Successfully requested the `SubscriptionListLink` included in the `EndDevice` instance payload and received a conformant response for the `SubscriptionList` payload from the Server. Successfully processed the `SubscriptionList` payload to confirm the Server supported the Subscription mechanism for its `EndDevice` instance. The Server successfully responded to the Client request with the correct payload that belongs to the incoming href.
- **[C, S]** The Client successfully found the `FSAList` assigned to the Client `EndDevice` and verified the `FSAList` could be subscribed to.
- **[C, S]** The Client successfully prepared and requested a subscription request to the Server using the correct href for the `NotificationURI` for each.
- **[S]** Successfully received and processed the subscription request and responded with a new Location header and HTTP `201 Created`.
- **[S]** Caused a change for one (or more) of the FSA included resource and sent a notification message with the updated resource payload as the Notification body.

- **[C, S]** The Client successfully received (respond with HTTP `201 Created` or HTTP `204 No Content`) the Notification message from the Server and processed the change in its internal state reflecting the updated resource information. Client may need to take further action based on the updated resource information sent. For example, a new `DERProgram` and/or `DERControl` was included.
- **[C, S]** The Client successfully sent the HTTP GET request on the resource included in the step 7 notification body from the Server. Successfully received the response from the Server and compared the response payload resource to the resource included in the previous step notification body and concluded they were identical.

CORE-019 - Advanced Subscription [A, S]

Purpose

Verify provisioning, acquisition, and execution of more complex subscription resources.

The basic subscription test verifies:

- The IEEE 2030.5 server supports a set of resources which can be subscribed to.
- The client devices can receive and act on notifications from those resources.

This test shall include advanced test scenarios, where notifications might be done on multiple conflicting resources to observe client handling of the changes.

Setup

1. **[S]** The `EndDevice` resource with the `SFDI/LFDI` value of the Client device shall have a `SubscriptionListLink` element included with the appropriate attributes so the Server and Client can use the `SubscriptionListLink` to support subscription and notification requests. This link shall be empty at this stage of this test. For example, `all=0`. This `EndDevice` instance shall be able to be subscribed to with value of one, unconditional, subscription.
2. **[S]** The `FSAList` that belongs to the `EndDevice` instance with the Client `SFDI/LFDI` shall be able to be subscribed to with value of one, unconditional, subscription.
3. **[S]** The server shall be able to update an attribute included in the Client `EndDevice` and FSA instances to trigger a notification message to the Client.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C, S]** Search the retrieved list of `EndDevice` instances from the Server for the Client `SFDI/LFDI` value. Next, find the `SubscriptionListLink` elements and its attribute to confirm the Server supports subscription. Confirm this `EndDevice` instance itself can be subscribed to.
3. **[C]** Prepare a subscription request for this `EndDevice` instance with required elements/attributes and send an HTTP POST request to the Server
4. **[S]** Receive the sent subscription request from step 3 and process the request and respond with an HTTP `201 Created` or HTTP `204 No Content` response and the correct URI location for the created resource for the subscription request stored on the Server.
5. **[C]** Search the `FSAList` assigned to the Client `EndDevice` and traverse through the found FSA instance for an element that can be subscribed to with attribute of one, as configured during test setup, above.
6. **[C]** For the FSA, which includes an attribute that can be subscribed to, prepare a subscription request with all required elements/attributes and send an HTTP POST request.

7. **[S]** For the received subscription request from the Client, process and send an HTTP `201 Created` or HTTP `204 No Content` response, after validating such a request can be made by the Client, with the correct HTTP location header, which indicates the URI of the created resource for the subscription request stored on the Server.
8. **[S]** Cause a change for one the elements of the FSA resource for the Client, which shall cause a notification message. At such resource change, send a notification message using the URI information sent by the Client from the previous test step and include the updated FSA resource payload.
9. **[C]** When the Notification message is received from the Server, process the incoming Notification message and payload, including validation and respond back to the Notification message with HTTP `201 Created` or HTTP `204 No Content` message.
10. **[S]** The resource change from step 8 shall also cause a related Notification message for the `EndDevice` subscribed to by the Client. The Server shall send a notification message using the URI information sent by the Client from the earlier `EndDevice` Subscription test step and include the updated FSA resource payload.
11. **[C]** When the Notification message is received from the Server for the `EndDevice` resource update, process the incoming Notification message and payload, including validation and respond back to the Notification message with HTTP `201 Created` or HTTP `204 No Content` message.
12. **[C]** Perform an HTTP GET request on the href of the resource included in the Server notification message. Process the payload returned from the Server in response to the HTTP GET request and verify that it is identical to the notification body payload. Repeat the HTTP GET and validation steps for each notification received.
13. **[S]** Cancel the outstanding FSA subscription by sending a notification message to Client indicating notification `status = 1` (Subscription canceled, no additional information) and required elements/attributes for the Notification payload.
14. **[C]** Receive the Notification message that indicates cancellation sent by the Server, process the message, and update the internal subscription state so future subscription requests are not done, or the Client continues to expect future Notifications on the canceled resource.

Pass/Fail Criteria

- **[C, S]** The Client successfully searched the retrieved list of `EndDevice` instances from the Server and found the correct instance, which includes the Client `SFDI/LFDI` value. Successfully requested the `SubscriptionListLink` included in the `EndDevice` instance payload and received a conformant response for the `SubscriptionList` payload from the Server. Successfully processed the `SubscriptionList` payload to confirm the Server supported the Subscription mechanism for its `EndDevice` instance. The Server successfully responded to the Client request with the correct payload that belongs to the incoming href.

- **[C, S]** The Client successfully found the `FSAList` assigned to the Client `EndDevice` and verified the `FSAList` could be subscribed to.
- **[C]** The Client successfully prepared and requested a subscription request to the Server using the correct href for the `NotificationURI` for each.
- **[S]** Server successfully received and processed the subscription request and responded with a new Location header and HTTP `201 Created`.
- **[S]** Server caused a change for one (or more) of the FSA included resource and sent a notification message with the updated resource payload as the Notification body.
- **[C]** The Client successfully received (respond with HTTP `201 Created` or HTTP `204 No Content`) the Notification message from the Server and processed the change in its internal state reflecting the updated resource information.
- **[C]** The Client successfully received the Notification cancellation message from the Server and provided an HTTP response.

CORE-021 - Randomized Events [C, A, S]

Purpose

Verify provisioning, acquisition, and execution of randomized events.

The randomized events test verifies the Client ability to handle DER events randomized to minimize the impact on overall grid stability. This test shall include DER events with plus/minus randomization in the event start and duration times.

Setup

1. **[S]** Construct 1 `DERProgramList`, which includes three `DERPrograms` and three `DERControl`, and required subordinate resources:
    - `DERProgram`1/`DERControl`1 has at minimum of one immediate control mode. It shall start three minutes from now with duration of two minutes and have highest Primacy value (lowest priority). This `DERControl`1 shall have `0` values for `randomizeStart` and `randomizeDuration` values.
    - `DERProgram`#2/`DERControl`#2 includes one curve-based control relevant for the target Client. It shall start six minutes from now with duration of one minute and have next highest Primacy value (second highest priority). This `DERControl`#2 shall have plus 30 seconds for `randomizeStart` and `randomizeDuration` values.
    - `DERProgram`#3/`DERControl`#3 includes an immediate control, which is not applicable for the Client and curve-based controls, which are relevant. It shall start nine minutes from now with duration of three minutes and have lowest Primacy value (highest priority). This `DERControl`#3 shall have minus 30 seconds for `randomizeStart` and `randomizeDuration` values.
    - At least one `DERProgram` shall have `DefaultDERControl` attribute and other requirements listed above.
    - Immediate- and curve-based controls shall conform to the above requirements.
    - All three `DERControl` instances shall be used across `DERProgram` and/or `DefaultDERControl` instances.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Process the received `DERProgramList` assigned to the Client through the FSA instance and find the highest priority `DERProgram` /`DERControl` to process by following the list ordering rules. The DER-Client shall schedule the selected `DERProgram/DERControl` with the correct `randomizeStart` and `randomizeDuration` values applied (should be `DERProgram` 3, above).

3. **[C]** Client shall issue subsequent HTTP GET requests to retrieve the subordinate resources associated with all of the `DERProgram` and `DERControl` instances, such as `DERCurveListLink` with one or more `DERCurveList` instances.
4. **[C]** For the `DERCurve` resources in the `DERCurveList`, the Client shall do HTTP GET requests to retrieve the 10 curve points and their values. The Client shall use these curve point and related values to construct the DER curve-based control, which shall be communicated to the internal inverter system and the schedule information. For example, start time + duration + `randomizationStart` + `randomizeDuration`.
5. **[C]** Until the selected `DERProgram/DERControl` becomes active, the Client shall use the `DefaultDERControl` value by setting the DER control mode using the `DefaultDERControl` attribute.
6. **[C]** Periodically poll the associated `DERControl` resource to check the Status of the DER event. Client completes the DER event based on either the `DERControl` Status value and/or start time + duration + `randomizationStart` + `randomizeDuration` period.
7. **[C]** Repeat steps #4 and 5 for rest of the `DERProgram/DERControl` instances that have are scheduled.

Pass/Fail Criteria

- **[C, S]** The Client successfully received a conformant, or any, `DERProgramList` payload from the Server. Client was able to find which multiple `DERPrograms` take priority by using the list ordering rules. Client was able to schedule a DER event based on the `DERProgram/DERControl` information, including randomization values applied, which was received.
- **[C, S]** The Client successfully retrieved the subordinate resources for the `DERProgram/DERControl` payload. For example, `DERCurve`List information. The Server sent all requested resource information as a response to the Client HTTP GET requests.
- **[C, S]** The Client successfully retrieved all curve points and their values from the Server. The Server sent all curve points and their values as requested by the Client. The Client successfully constructed the curve points and their values to send the information to the DER inverter system.
- **[C]** The Client successfully found and scheduled the `DefaultDERControl` while no DER event was active.
- **[C]** The Client successfully retrieved the `DERControl` status to find the completion of the DER event and successfully completed the DER Event, including the application of the randomization values.

CORE-022 – Responses [C, A, S]

Purpose

Verify generation and handling of event responses.

The responses test verifies the Client ability to handle responses to programs scheduled by the Server. This test shall include DERPrograms that specify responses by the Client for various DER event states.

Setup

1. **[S]** Construct 1 DERProgramList, which includes 3 DERPrograms and 3 DERControl (and required subordinate resources):
    - DERProgram1/DERControl1 has at minimum of one immediate control mode relevant for the target Client. It shall start three minutes from now with duration of two minutes and have highest Primacy value (lowest priority).
    - DERProgram#2/DERControl#2 includes one curve-based control relevant for the target Client. It shall start six minutes from now with duration of one minute and have next highest Primacy value (second highest priority).
    - DERProgram#3/DERControl#3 includes an immediate control, which is not applicable for the Client and curve-based controls, which are relevant. It shall start nine minutes from now with duration of three minutes and have lowest Primacy value (highest priority).
    - At least one DERProgram shall have DefaultDERControl attribute and other requirements listed above.
    - Immediate- and curve-based controls shall conform to the above requirements.
    - Each DERControl shall include a replyTo and responseRequired elements as follows:
        - The replyTo element of the DERControl resource shall map to the available URI location where the Server accepts POST of responses from the Client devices.
        - The responseRequired element of the DERControl resource shall be specified with the value of 7. See the IEEE 2030.5 standard response status code table, which requires Client devices to send responses at each state of the DERControl event.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Process the received DERProgramList assigned to the Client through the FSA instance and find the highest priority DERProgram /DERControl to process by following the list ordering rules. The DER-Client shall schedule the selected DERProgram/DERControl (should be DERProgram 3, above).

3. **[C]** After the correct DERProgram/DERControl is selected, Client shall issue subsequent HTTP GET requests to retrieve the subordinate resources associated with the DERProgram and DERControl, such as DERCurveListLink with one or more DERCurveList instances.
4. **[C]** For the DERCurve resources in the DERCurveListLink, the Client shall do HTTP GET requests to retrieve the 10 curve points and their values. The Client shall use these curve point and related values to construct the DER curve-based control, which shall be communicated to the internal inverter system and the schedule information. For example, start time and duration.
5. **[C]** Until the selected DERProgram#1/DERControl#1 becomes active, the Client shall use the DefaultDERControl value by setting the DER control mode using the DefaultDERControl attribute.
6. **[S]** Update the DERControl#1 currentStatus/dateTime values at each state of the DER event by following its event schedule. Before the event completes its schedule, cancel the event by updating its currentStatus value to 2 (canceled).
7. **[C]** Periodically poll the associated DERControl#1 resource to check the Status of the DER event and send the requested response at the responseRequired attribute, such as a Response message with status=1 (Received) and status=2 (Started). Client completes the DER event either based on the DERControl#1 Status value and/or start time plus duration interval. Because this event was canceled in step 5, the Client shall notice such cancellation and send a response with status=6 (cancellation).
8. **[S]** Update the DERControl#2 currentStatus/dateTime values at each state of the DER event by following its event schedule.
9. **[C]** Periodically poll the associated DERControl#2 resource to check the Status of the DER event and send the requested response at the responseRequired attribute. Client completes the DER event either based on the DERControl#2 Status value and/or start time plus duration interval.
10. **[S]** Update the DERControl#3 currentStatus/dateTime values at each state of the DER event by following its event schedule.
11. **[C]** Periodically poll the associated DERControl#3 resource to check the Status of the DER event and send the requested response at the responseRequired attribute. Client completes the DER event either based on the DERControl#3 Status value and/or start time plus duration interval.

Pass/Fail Criteria

- **[C, S]** The Client successfully received a conformant, or any, DERProgramList payload from the Server. Client was able to find which multiple DERPrograms take priority by using the list ordering rules. Client was able to schedule a DER event based on the DERProgram/DERControl information, including randomization values applied, which was received.

- **[C, S]** The Client successfully retrieved the subordinate resources for the `DERProgram/DERControl` payload. For example, `DERCurve`List information. The Server sent all requested resource information as a response to the Client HTTP GET requests.
- **[C, S]** The Client successfully retrieved all curve points and their values from the Server. The Server sent all curve points and their values as requested by the Client. The Client successfully constructed the curve points and their values to send the information to the DER inverter system.
- **[S]** Server successfully updated the `currentStatus`/`dateTime` attributes at the event states, including when the event was canceled.
- **[C]** The Client successfully found and scheduled the `DefaultDERControl` while no DER event was active.
- **[C, S]** The Client successfully retrieved the `DERControl#1` status to detect the event status from its creation to cancellation including sending the Response messages that correspond to: Status=1 (received), Status=2 (started), Status = 6 (cancellation) at the correct times.
- **[C, S]** The Client successfully retrieved the `DERControl#2` and `DERControl#3` status to detect the event status from its creation to completion including sending the Response messages that correspond to: Status=1 (received), Status=2 (started), Status = 3 (completed) at the correct times.

7 Basic Functions Tests

This section specifies the tests for the CSIP IEEE 2030.5 Basic Functions section. These tests include:

- DER identification/commissioning of inverters
- Group management
- Inverter curve and immediate based controls
- Event prioritization of operations and conflict resolution
- Basic communication interactions
- Schedule of future operations
- Standard inverter status and alarm iInformation

These tests ensure that the client and server implementations conform to CSIP basic functions and are a prerequisite for inverter/CSIP-related tests specified in subsequent sections of this document.

BASIC-001 - DER Identification [C, A, S]

Purpose

Verify provisioning, acquisition, and accuracy of SFDI, LFDI, and pIN values. Verify provisioning, and acquisition of end device resources.

The DER identification test verifies that the Clients commissioned by the server generate unique IEEE 2030.5 `SFDI/LFDI` IDs and validate registered PIN values.

The commission process is out of scope for this test.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. On the Server and Client device(s): Confirm the TLS certificates are authorized and installed so the TLS 1.2 based communication as required by the IEEE 2030.5 standard can occur for this test. `SFDI` and `LFDI` numbers are hashed values of the TLS certificates.
3. **[S]** Pre-register at least two `EndDevice` instances, including an instance for the Client device itself, including all following attributes. For example, `SFDI`, `LFDI`, `FunctionSetAssignmentsListLink`, `RegistrationLink`/`PIN`. The `FunctionSetAssignments` shall include `DERProgram` and its subordinate resources for the `EndDevice` instances.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Retrieve the `DeviceCapability` resource from the Server using the supported HTTP and IP address and find the `EndDeviceListLink` element.
3. **[C]** Perform an HTTP GET operation on the `EndDeviceListLink` URI and search through the `EndDeviceList` payload to find if an `EndDevice` instance is included that matches the identity of the Client device. For example, `SFDI/LFDI`. If found, continue to next step. A valid instance must be found because such instance was preregistered during test setup. If none is found, raise an error for the Server and stop this test.

4. **[C]** Using the `EndDevice` instance found from the previous step, validate the `SFDI` and `LFDI` values by:
    - The `SFDI` value presented by the Server through the matching `EndDevice` instance for the Client shall match the `SFDI` value the Client itself has computed using its own TLS certificates. `SFDI` is a 36-bit left-truncated value of the SHA256 hash of the TLS certificate where it is expressed as 11 decimal digits with last digit being a check digit. See IEEE 2030.5, section 8.3, Device Credentials.
    - The `LFDI` value presented by the Server through the matching `EndDevice` instance for the Client shall match the `LFDI` value the Client itself has computed using its own TLS certificates. `SFDI` is a 16-bit left-truncated value of the SHA256 hash of the TLS certificate where it is expressed as 40 hexadecimal digits. See IEEE 2030.5, section 8.3, Device Credentials.
5. **[C]** Using the RegistrationLink/PIN instance found from the EndDevice instance, validate the PIN values by:
    - The PIN value presented by the Server through the matching RegistrationLink/PIN instance for the Client shall match the PIN value the Client itself has registered. PIN is a 6-digit integer value with last digit being a check digit. See IEEE 2030.5, section 8.2.1 Local registration attributes.
6. **[C]** For an aggregator client, iterate through rest of the `EndDevice` instances included in the original `EndDeviceList` from the previous test step and validate each set of `SFDI`/`LFDI` and `PIN` values. In this case, the aggregator type of device shall already know the identities of these devices, including their `SFDI`/`LFDI`/`PIN` and other device-specific values to validate these entries sent by the Server.
7. **[C]** Using the `FunctionSetAssignmentsListLink` from test step 4, perform an HTTP GET operation on the included resource inside the `FunctionSetAssignmentsList` the link refers to. Follow the resource subordinate resources by performing subsequent HTTP GET operations on them.

Pass/Fail Criteria

- **[C, S]** Client requested and received the `DeviceCapability` resource on the Server using the HTTP configuration information provided. Server responded with `200 OK` and returned a conformant payload for its `DeviceCapability`.
- **[C, S]** Server returned an `EndDeviceList` payload in response to the Client HTTP GET request. This `EndDeviceList` payload returned from the Server shall include the Client `EndDevice` instance and any other associated `EndDevice` instances the Client manages. For example, as an aggregator client.
- **[C, S]** Client validated the correctness of the `SFDI`, `LFDI`, and `RegistrationLink/PIN` values included in the `EndDevice` instance assigned to it.

The `SFDI`, `LFDI`, and `RegistrationLink/PIN` related testable requirements must be met during this validation step.

- **[C, S]** The Client successfully traversed through the resources included in its `EndDevice` instance payload and successfully received and validated the requested payload through the series of HTTP GET operations. Server successfully received the HTTP GET request from the Client and responded back with successfully responses for the request, including XML conformant payload for each request.

BASIC-002 - Basic Group Management [C, A, S]

Purpose

Verify provisioning and acquisition of end device with a seven-level resource hierarchy as specified in Figure 2.

The basic group management test verifies inverter grouping based on the CSIP topology concept, and the Client ability to interpret the created groups and make use of the assigned resources.

Setup

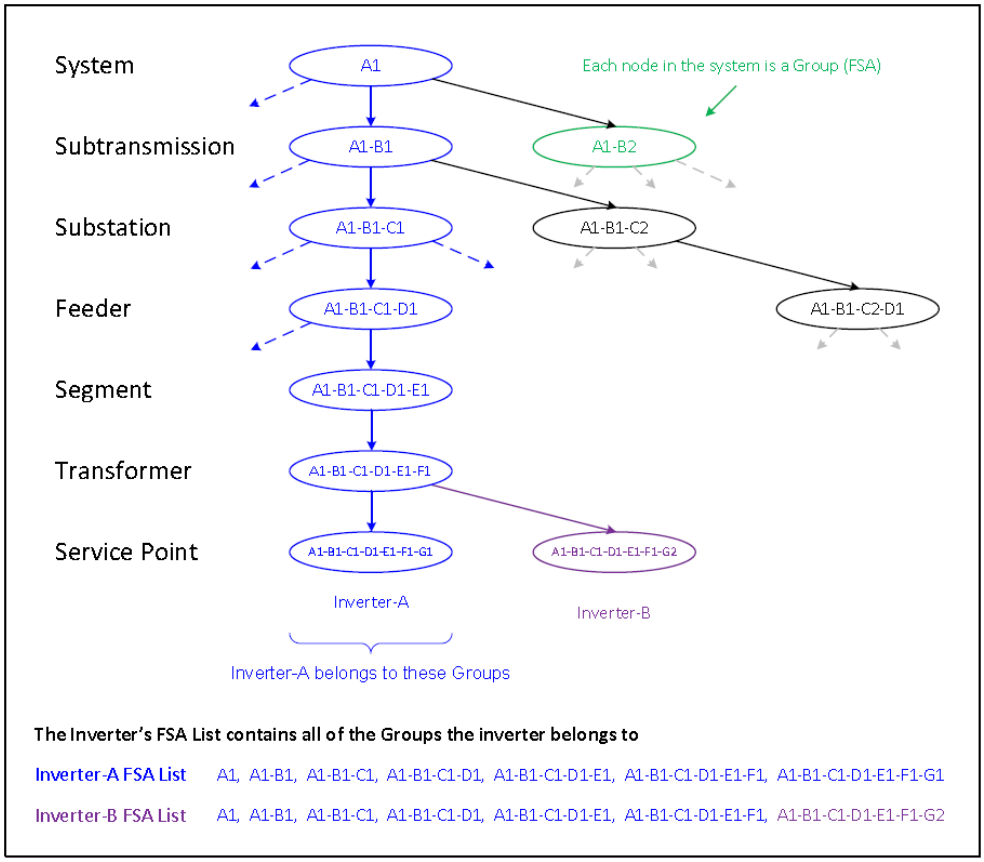

Figure 2 Basic Group Management Test Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in the above diagram. Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
   - `SFDI`

Short form device identifier. One of the three EndDevice instances must have an SFDI that matches the SFDI of the Client.

- LFDI
  Long form device identifier. One of the three EndDevice instances must have an LFDI that matches the LFDI of the Client.
- RegistrationLink
  The link to the Registration resource of the EndDevice instance with the Registration:PIN resource set to 111115.
- FunctionSetAssignmentsListLink
  The link to the FunctionSetAssignmentsList resource of the EndDevice instance.

4. **[S]** Construct a FunctionSetAssignmentsList resource for the Client with seven FunctionSetAssignments following the above example diagram. The closest node to the inverter itself (Service Point) shall have a DERProgram assigned to it with the highest priority (lowest primacy) and earliest start time (starts in plus-five minutes) in its DERControl instance with a valid immediate or curve-based control(s). At least one DERProgram shall also have a non-empty DefaultDERControl resource associated with it so the DER device can activate such control during idle periods. All other DERProgram shall have an empty DERControlList.

Procedure

1. **[T]** Record the Client/Server communications.
2. The Client shall have retrieved all of its seven FunctionSetAssignments that correspond to the above grid topology-based diagram to continue execution of this test.
3. **[C]** Using the FSA instances retrieved from the CORE-010 - Function Set Assignments test, process the elements in the FSA instance to do additional HTTP GET requests for all of its subordinate resources. Validate there is a Time resource assigned to the FSA instance.
4. **[C]** Do HTTP GET requests on the DERProgramListLink and TimeLink hrefs for the FSA instance using l=255 query string search parameter.
5. **[C]** Process the information in the returned DERProgramList from the previous step and find which DERProgram has the highest priority based on IEEE 2030.5 event priority determination rules.
6. **[C]** Process the contents of the highest priority DERProgram and find which DERControl to make active: DefaultDERControl or DERControlListLink associated DERControl. Do additional HTTP GET requests for subordinate resources to find which DERControl to make active.
7. **[C]** Based on the determination in the previous step, make the appropriate DERControl active. Based on the test setup, it shall be the DefaultDERControl that becomes active.

8. **[C]** Periodically poll the FSA resource for updated information in the FSA instance.

Pass/Fail Criteria

- **[C, S]** The Client successfully processed all FSA instances retrieved and verified there is a valid Time resource assigned in the FSA instance.
- **[C, S]** The Client successfully issued HTTP GET requests on `DERProgramListLink` or `TimeLink` from the Server and received conformant responses for each.
- **[C, S]** The Client successfully found which `DERProgram` instance had highest priority. The Server returned `DERProgram` instances that permitted the Client to find which had highest priority to take action on.
- **[C, S]** The Client successfully processed the contents of the highest priority `DERProgram` and found which `DERControl` to make active: `DefaultDERControl`. Successfully did additional HTTP GET requests for subordinate resources to retrieve the `DERControl` associated with the `DefaultDERControl`.
- **[C, S]** The Client successfully found and made the `DefaultDERControl` active.
- **[C, S]** The Client successfully polled and received the FSA resource for updated information in the FSA instance each time.

BASIC-003 - Advanced Group Management [C, A, S]

Purpose

Verify provisioning and acquisition of end device with a seven-level resource hierarchy and changes applied to the hierarchy after initial acquisition.

The advanced group management test verifies advanced inverter grouping based on the CSIP topology concept, and the Client ability to interpret the created groups and make use of the assigned resources.

Setup

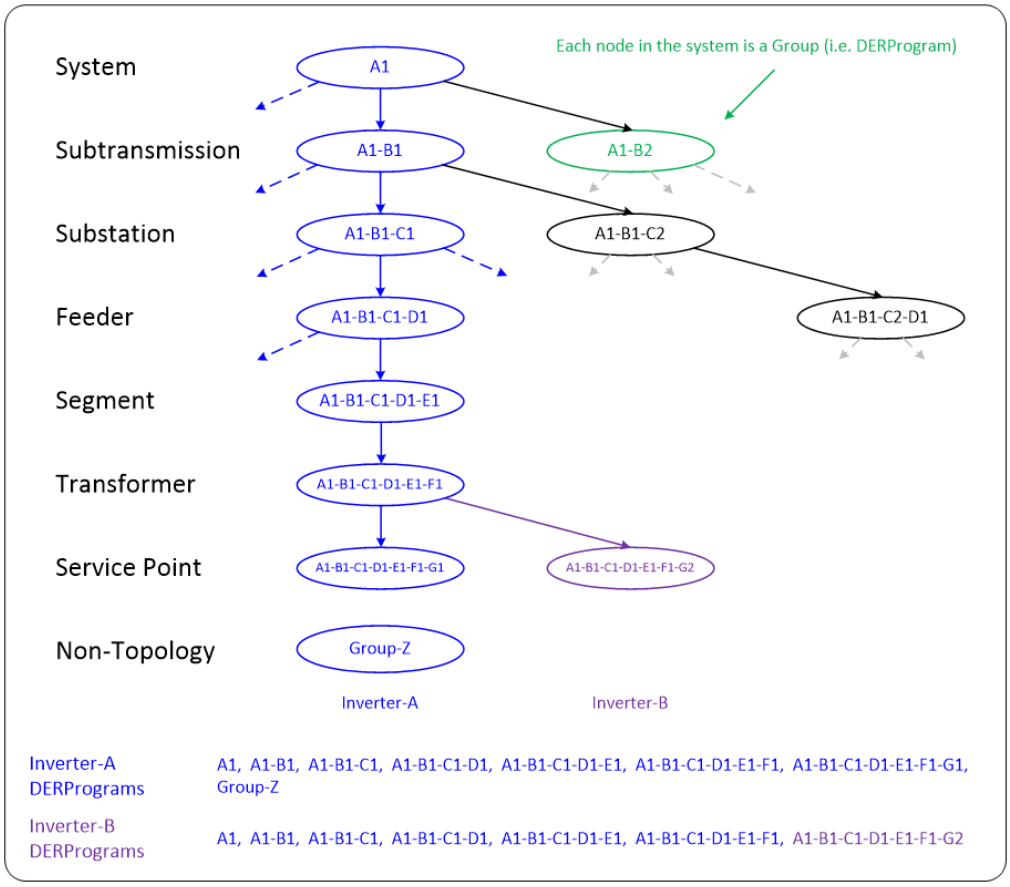

Figure 3 Advanced Group Management Test Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
3. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in the above diagram. Make sure the

changedTime of the EndDevice instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.

4. **[S]** For each EndDevice, these fields must be populated:
   - SFDI
     Short form device identifier. One of the three EndDevice instances must have an SFDI that matches the sFDI of the Client.
   - LFDI
     Long form device identifier. One of the three EndDevice instances must have an LFDI that matches the LFDI of the Client.
   - RegistrationLink
     The link to the Registration resource of the EndDevice instance with the Registration:PIN resource set to 111115.
   - FunctionSetAssignmentsListLink
     The link to the FunctionSetAssignmentsList resource of the EndDevice instance.
5. **[S]** Construct a FunctionSetAssignmentsList resource for the Client with seven FunctionSetAssignments following Figure 3. The second closest node to the inverter itself, Transformer, shall have a DERProgram assigned to it with the highest priority (lowest primacy value) and earliest start time (starts in plus-fifteen minutes and lasts for five minutes) in its DERControl instance with a valid immediate or curve-based control(s). At least one DERProgram shall also have a non-empty DefaultDERControl resource associated with it so the DER device can activate such control during idle periods. Other DERPrograms for other nodes in the topology shall have lower priorities.

Procedure

1. **[T]** Record the Client/Server communications.
2. The Client shall have retrieved all of its seven FunctionSetAssignments that correspond to the topology specified in Figure 3 to continue execution of this test.
3. **[S]** Before the scheduled DERProgram/DERControl starts, change the FSAList assigned to this Client by removing the specific feeder in the above diagram and adding a new feeder to the topology. For example, A1-B1-C1-D2 instead of A1-B1-C1-D1. This also causes removal of the nodes below tied to the A1-B1-C1-D1 feeder. Add new nodes based on the new feeder and assign a different DefaultDERControl instance and DERControl (Effective start time at plus-ten minutes for five minutes duration).
4. **[C]** Poll the FSA resource for updated information in the FSA instance, which shall cause the Client to notice the updated FSAList and issue subsequent HTTP GETs to retrieve the updated subordinate information.
5. **[C]** Based on the updated resources from the previous step, schedule a different DefaultDERControl included in the updated FSAList DERProgram.

Pass/Fail Criteria

- **[S]** Server successfully updated the `FSAList` for the Client to communicate the Feeder node level-based changes.
- **[C, S]** The Client successfully polled and detected the updated `FSAList` information and retrieved all subordinate resources. The Server successfully received and responded to the polling requests and subordinate resource HTTP GET requests.
- **[C]** The Client successfully processed and prioritized the correct `DefaultDERControl` included in the highest priority `DERProgram` and made such `DefaultDERControl` active until the scheduled `DERControl` event becomes active.

BASIC-004 - Basic Inverter Control (Low/High Voltage Ride-Through) [C, A, S]

Purpose

Verify provisioning, acquisition, and execution of low/high voltage ride-through controls.

The basic inverter control test verifies the inverter ability to process immediate and curve-based controls supported by the IEEE 2030.5 standard and specified in the CSIP specification.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in Figure 3). Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
    - `SFDI`
      Short form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the Client.
    - `LFDI`
      Long form device identifier. One of the three `EndDevice` instances must have an `LFDI` that matches the `LFDI` of the Client.
    - `RegistrationLink`
      The link to the Registration resource of the `EndDevice` instance with the `Registration:PIN` resource set to `111115`
    - `FunctionSetAssignmentsListLink`
      The link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
4. **[S]** Construct a `FunctionSetAssignmentsList` resource for the Client with two `FunctionSetAssignments` (topology and non-topology FSAs) following Figure 3. The `DERPrograms` for the topology `FunctionSetAssignment` instance for the Client shall be configured as follows:
    - Closest node/FSA to the DER device (Service Point) shall include a `DERProgram` with the highest priority (lowest primacy value).
    - At least one `DERPrograms` shall have a `DefaultDERControl` with a valid DER inverter control the DER device must activate during idle periods.
    - The Service Point `DERProgram` shall have a `DERControl` instance with Low/High Voltage Ride Through values in Figure 4 Low/High Voltage Trip Settings.

- The DERControl shall have an effective start time of (plus-two minutes) and duration (plus-one minute) specified. The DERControl shall have the following elements:
    - The replyTo element of the DERControl resource shall map to the available URI location where the Server accepts POST of responses from the Client devices.
    - The responseRequired element of the DERControl resource shall be specified with the value of 7. See the IEEE 2030.5 standard response status code table, which requires Client devices to send DERControlResponses at each state of the DERControl event.
- Low/High Voltage Ride Through DERControl must be tested as specified in the CSIP specification.

Function	IEEE 2030.5 Control	Control Type
Low/High Voltage Ride Through	opModLVRTMustTrip opModLVRTMomentaryCessation opModHVRTMustTrip opModHVRTMomentaryCessation	Curve

Setting	Default	Test Values
opModHVRTMustTrip.DERCurve.CurveData	(16, 13000), (16, 12000), (1300, 12000), (1300, 11000), (10000, 11000)	(16, 12000), (16, 11000), (1200, 11000), (1200, 10000), (10000, 10000)
opModHVRTMustTripMomentaryCessation.DERCurve.CurveData	(0, 11000), (1300, 11000)	(0, 10000), (1200, 10000)
opModLVRTMustTrip.CurveData	(150,0),(150,5000),(1100,5000),(1100,7000),(2100,7000), (2100, 8800), (10000, 8800)	(150,0),(150,5000),(1200,5000),(1200,7000),(2200,7000), (2200, 8800), (10000, 8800)
opModLVRTMustTripMomentaryCessation.DERCurve.CurveData	(0, 5000), (150, 5000)	(0, 6000), (150, 6000)
DERCurve.curveType	4, 5, 9, 10	4,5,9,10
DERCurve.xMultiplier	-2	-2
DERCurve.yMultiplier	-2	-2
DERCurve.yRefType	4	4

Figure 4 Low/High Voltage Trip Settings

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** The Client shall have retrieved all its FSA group and associated resources, including all subordinate resources for DERProgram, DERControl, and DefaultDERControl,

DERCurveList and others. The Client would have processed all resource information for the FSA to find the priorities of each DERProgram and associated DERControl resources. Now, the two DERProgram and DERControl would have been scheduled, as specified in the test setup, and polling the FSAList for updated information, if any.

3. **[C]** Using the DefaultDERControl information associated with the highest priority DERProgram, the Client shall activate the DefaultDERControl until the Service Point DER event becomes active.
4. **[C]** At the effective start time of the highest priority DERProgram/DERControl, it shall check the currentStatus of the DERControl by executing an HTTP GET on the selected DERControl href. After the currentStatus field is verified, Client shall activate this DERControl using the subordinate resource information, including the immediate or curve-based control required. If a response is required, Client shall send a series of response message based on the response requirements specified in the DERControl.
5. **[S]** Process the sent response messages and verify they reflect the currentStatus of the actual DER event.
6. **[C]** Complete the current DERControl event at the effective end time and send the required response message to the Server. If there is no other active event, it defaults to the DefaultDERControl information associated with the highest priority DERProgram.
7. **[S]** Verify all the immediate or curve-based control is exercised in this test for the Client by checking the response message sent by the Client for each DERControl event.

Pass/Fail Criteria

- **[C, S]** The Client successfully detected and activated the DefaultDERControl specified by the Server for the highest priority DERProgram/DERControl.
- **[C, S]** The Client successfully checked the currentStatus before activating the DERControl event. The Client successfully activated the correct inverter control mode (Low/High Voltage Ride Through) specified in the DERControl. The Client successfully sent a series of response message based on the response requirements specified in the DERControl.
- **[C, S]** Server successfully processed response messages sent by the Client and verified they reflect the currentStatus of the actual DER event.
- **[C, S]** The Client successfully completed the current DERControl event at the effective end time and sent the required response message to the Server.

BASIC-005 - Basic Inverter Control (Low/High Frequency Ride-Through) [C, A, S]

Purpose

Verify provisioning, acquisition, and execution of low/high frequency ride-through controls

The basic inverter control test verifies the inverter ability to process immediate and curve-based controls supported by the IEEE 2030.5 standard and specified by the CSIP specification.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in Figure 3). Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
    - `SFDI`
      Short form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the Client.
    - `LFDI`
      Long form device identifier. One of the three `EndDevice` instances must have an `LFDI` that matches the `LFDI` of the Client.
    - `RegistrationLink`
      The link to the Registration resource of the `EndDevice` instance with the `Registration:PIN` resource set to `111115`
    - `FunctionSetAssignmentsListLink`
      The link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
4. **[S]** Construct a `FunctionSetAssignmentsList` resource for the Client with two `FunctionSetAssignments` (topology and non-topology FSAs) following Figure 3. The `DERPrograms` for the topology `FunctionSetAssignment` instance for the Client shall be configured as follows:
    - Closest node/FSA to the DER device (Service Point) shall include a `DERProgram` with the highest priority (lowest primacy value).
    - At least one `DERPrograms` shall have a `DefaultDERControl` with a valid DER inverter control that the DER device must activate during idle periods.
    - The Service Point `DERProgram` shall have a `DERControl` instance with Low/High Frequency Ride Through values in Figure 5 Low/High Frequency Trip Settings.

- The DERControl shall have an effective start time of (plus-two minutes) and duration (plus-one minute) specified. The DERControl shall have the following elements:
    - The replyTo element of the DERControl resource shall map to the available URI location where the Server accepts POST of responses from the Client devices.
    - The responseRequired element of the DERControl resource shall be specified with the value of 7. See the IEEE 2030.5 standard response status code table, which requires Client devices to send DERControlResponses at each state of the DERControl event.

- Low/High Frequency Ride Through DERControl must be tested as specified in the CSIP specification.

Function	IEEE 2030.5 Control	Control Type
Low/High Frequency Ride Through	opModLFRTMustTrip opModHFRTMustTrip	Curve

Setting	Default	Test Values
opModHFRTMustTrip.DERCurve.CurveData	(16,6400),(16,6200),(30000, 6200),(30000,6050), (40000,6050)	(16,6500),(16,6100),(28000, 6100),(28000,6050), (40000,6050)
opModLFRTMustTrip.DERCurve.CurveData	(16, 5300), (16,5690), (30000,5690), (30000, 5850), (40000, 5850)	(16, 5300), (16,5690), (27000,5690), (27000, 5850), (40000, 5850)
DERCurve.curveType	2, 7	2,7
DERCurve.xMultiplier	-2	-2
DERCurve.yMultiplier	-2	-2

Figure 5 Low/High Frequency Trip Settings

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** The Client shall have retrieved all its FSA group and associated resources, including all subordinate resources for DERProgram, DERControl, and DefaultDERControl, DERCurveList and others. The Client would have processed all resource information for the FSA to find the priorities of each DERProgram and associated DERControl resources. Now, the two DERProgram and DERControl would have been scheduled, as specified in the test setup, and polling the FSAList for updated information, if any.
3. **[C]** Using the DefaultDERControl information associated with the highest priority DERProgram, the Client shall activate the DefaultDERControl until the Service Point DER event becomes active.

4. **[C]** At the effective start time of the highest priority `DERProgram/DERControl`, it shall check the `currentStatus` of the `DERControl` by executing an HTTP GET on the selected `DERControl` href. After the `currentStatus` field is verified, Client shall activate this `DERControl` using the subordinate resource information, including the immediate or curve-based control required. If a response is required, Client shall send a series of response message based on the response requirements specified in the `DERControl`.
5. **[S]** Process the sent response messages and verify they reflect the `currentStatus` of the actual DER event.
6. **[C]** Complete the current `DERControl` event at the effective end time and send the required response message to the Server. If there is no other active event, it defaults to the `DefaultDERControl` information associated with the highest priority `DERProgram`.
7. **[S]** Verify all the immediate or curve-based control is exercised in this test for the Client by checking the response message sent by the Client for each `DERControl` event.

Pass/Fail Criteria

- **[C, S]** The Client successfully detected and activated the `DefaultDERControl` specified by the Server for the highest priority `DERProgram/DERControl`.
- **[C, S]** The Client successfully checked the `currentStatus` before activating the `DERControl` event. The Client successfully activated the correct inverter control mode (Low/High Frequency Ride Through) specified in the `DERControl`. The Client successfully sent a series of response message based on the response requirements specified in the `DERControl`.
- **[C, S]** Server successfully processed response messages sent by the Client and verified they reflect the `currentStatus` of the actual DER event.
- **[C, S]** The Client successfully completed the current `DERControl` event at the effective end time and sent the required response message to the Server.

BASIC-006 - Basic Inverter Control (Volt/Var) [C, A, S]

Purpose

Verify provisioning, acquisition, and execution of volt-var control.

The basic inverter control test verifies the inverter ability to process immediate and curve-based controls supported by the IEEE 2030.5 standard and specified by the CSIP specification.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in Figure 3). Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
    - `SFDI`
      Short form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the Client.
    - `LFDI`
      Long form device identifier. One of the three `EndDevice` instances must have an `LFDI` that matches the `LFDI` of the Client.
    - `RegistrationLink`
      Link to the Registration resource of the `EndDevice` instance with the `Registration:PIN` resource set to `111115`
    - `FunctionSetAssignmentsListLink`
      Link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
4. **[S]** Construct a `FunctionSetAssignmentsList` resource for the Client with two `FunctionSetAssignments` (topology and non-topology FSAs) following Figure 3. The `DERPrograms` for the topology `FunctionSetAssignment` instance for the Client shall be configured as follows:
    - Closest node/FSA to the DER device (Service Point) shall include a `DERProgram` with the highest priority (lowest primacy value).
    - At least one `DERPrograms` shall have a `DefaultDERControl` with a valid DER inverter control the DER device must activate during idle periods.
    - The Service Point `DERProgram` shall have a `DERControl` instance with Volt-VAr values in Figure 6 Volt-VAr Settings.
    - The `DERControl` shall have an effective start time of (plus-two minutes) and duration (plus-one minute) specified. The `DERControl` shall have the following elements:

- The `replyTo` element of the `DERControl` resource shall map to the available URI location where the Server accepts POST of responses from the Client devices.
- The `responseRequired` element of the `DERControl` resource shall be specified with the value of 7. See the IEEE 2030.5 standard response status code table, which requires Client devices to send DERControlResponses at each state of the `DERControl` event.

- Volt-VAr DERControl must be tested using values as specified in the CSIP specification.

Function	IEEE 2030.5 Control	Control Type
Volt-VAr	`opModVoltVar`	Curve

Setting	Default	Test Values
opModVoltVar.DERCurve.CurveData	(9200, 3000),(9670, 0),(10300, 0),(10700, -3000)	(9100, 4000),(9570, 0),(10400, 0),(10600, -4000)
opModVoltVar.DERCurve.openLoopTms	10	5
opModVoltVar.DERCurve.autonomousVrefEnable	false	false
opModVoltVar.DERCurve.autonomousVrefTimeContant	0	0
DERCurve.curveType	11	11
DERCurve.xMultiplier	-2	-2
DERCurve.yMultiplier	-2	-2

Figure 6 Volt-VAr Settings

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** The Client shall have retrieved all its FSA group and associated resources, including all subordinate resources for `DERProgram`, `DERControl`, and `DefaultDERControl`, `DERCurveList` and others. The Client would have processed all resource information for the FSA to find the priorities of each `DERProgram` and associated `DERControl` resources. Now, the two `DERProgram` and `DERControl` would have been scheduled, as specified in the test setup, and polling the `FSAList` for updated information, if any.
3. **[C]** Using the `DefaultDERControl` information associated with the highest priority `DERProgram`, the Client shall activate the `DefaultDERControl` until the Service Point DER event becomes active.

4. **[C]** At the effective start time of the highest priority `DERProgram/DERControl`, it shall check the `currentStatus` of the `DERControl` by executing an HTTP GET on the selected `DERControl` href. After the `currentStatus` field is verified, Client shall activate this `DERControl` using the subordinate resource information, including the immediate or curve-based control required. If a response is required, Client shall send a series of response message based on the response requirements specified in the `DERControl`.
5. **[S]** Process the sent response messages and verify they reflect the `currentStatus` of the actual DER event.
6. **[C]** Complete the current `DERControl` event at the effective end time and send the required response message to the Server. If there is no other active event, it defaults to the `DefaultDERControl` information associated with the highest priority `DERProgram`.
7. **[S]** Verify all the immediate or curve-based control is exercised in this test for the Client by checking the response message sent by the Client for each `DERControl` event.

Pass/Fail Criteria

- **[C, S]** The Client successfully detected and activated the `DefaultDERControl` specified by the Server for the highest priority `DERProgram/DERControl`.
- **[C, S]** The Client successfully checked the `currentStatus` before activating the `DERControl` event. The Client successfully activated the correct inverter control mode (Dynamic Volt-VAr) specified in the `DERControl`. The Client successfully sent a series of response message based on the response requirements specified in the `DERControl`.
- **[C, S]** Server successfully processed response messages sent by the Client and verified they reflect the `currentStatus` of the actual DER event.
- **[C, S]** The Client successfully completed the current `DERControl` event at the effective end time and sent the required response message to the Server.

BASIC-007 - Basic Inverter Control (Ramp Rates) [C, A, S]

Purpose

Verify provisioning, acquisition, and execution of ramp rates control.

The basic inverter control test verifies the inverter ability to process immediate and curve-based controls supported by the IEEE 2030.5 standard and specified by the CSIP specification.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in Figure 3). Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
    - `SFDI`
      Short form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the Client.
    - `LFDI`
      Long form device identifier. One of the three `EndDevice` instances must have an `LFDI` that matches the `LFDI` of the Client.
    - `RegistrationLink`
      Link to the Registration resource of the `EndDevice` instance with the `Registration:PIN` resource set to `111115`
    - `FunctionSetAssignmentsListLink`
      Link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
4. **[S]** Construct a `FunctionSetAssignmentsList` resource for the Client with two `FunctionSetAssignments` (topology and non-topology FSAs) following Figure 3. The `DERPrograms` for the topology `FunctionSetAssignment` instance for the Client shall be configured as follows:
    - Closest node/FSA to the DER device (Service Point) shall include a `DERProgram` with the highest priority (lowest primacy value).
    - At least one `DERPrograms` shall have a `DefaultDERControl` with a valid DER inverter control the DER device must activate during idle periods.
    - The Service Point `DERProgram` shall have a `DefaultDERControl` instance with ramp rate setting values (see table below).
    - Ramp rate settings (DefaultDERControl) must be tested as specified in the CSIP specification.

Function	IEEE 2030.5 Control	Control Type

Ramp Rate Setting	`setGradW` `setSoftGradW`	Default-Only

Setting	Default	Test Values
Default.DERControl.setGradW	10000 (100%)	9000
Default.DERControl.setSoftGradW	200 (2%)	400

Figure 7 Ramp Rates Settings

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** The Client shall have retrieved all its FSA group and associated resources, including all subordinate resources for `DERProgram`, `DERControl`, and `DefaultDERControl`, `DERCurve`List and others. The Client would have processed all resource information for the FSA to find the priorities of each `DERProgram` and associated `DERControl` resources. Now, the two `DERProgram` and `DERControl` would have been scheduled, as specified in the test setup, and polling the `FSAList` for updated information, if any.
3. **[C]** Using the `DefaultDERControl` information associated with the highest priority `DERProgram`, the Client shall activate the `DefaultDERControl` until the Service Point DER event becomes active.
4. **[S]** Verify all the immediate or curve-based control is exercised in this test for the Client by checking the response message sent by the Client for each `DERControl` event.

Pass/Fail Criteria

- **[C, S]** The Client successfully detected and activated the `DefaultDERControl` (Ramp rates) specified by the Server for the highest priority `DERProgram/DERControl`.

BASIC-008 - Basic Inverter Control (Fixed Power Factor) [C, A, S]

Purpose

Verify provisioning, acquisition, and execution of fixed power factor control.

The basic inverter control test verifies the inverter ability to process immediate and curve-based controls supported by the IEEE 2030.5 standard and specified by the CSIP specification.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in Figure 3). Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
    - `SFDI`
      Short form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the Client.
    - `LFDI`
      Long form device identifier. One of the three `EndDevice` instances must have an `LFDI` that matches the `LFDI` of the Client.
    - `RegistrationLink`
      Link to the Registration resource of the `EndDevice` instance with the `Registration:PIN` resource set to `111115`
    - `FunctionSetAssignmentsListLink`
      Link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
4. **[S]** Construct a `FunctionSetAssignmentsList` resource for the Client with two `FunctionSetAssignments` (topology and non-topology FSAs) following Figure 3. The `DERPrograms` for the topology `FunctionSetAssignment` instance for the Client shall be configured as follows:
    - Closest node/FSA to the DER device (Service Point) shall include a `DERProgram` with the highest priority (lowest primacy value).
    - At least one `DERPrograms` shall have a `DefaultDERControl` with a valid DER inverter control the DER device must activate during idle periods.
    - The Service Point `DERProgram` shall have a `DERControl` instance with fixed power factor control values in Figure 8 Fixed Power Factor Settings.
    - The `DERControl` shall have an effective start time of (plus-two minutes) and duration (plus-one minute) specified. The `DERControl` shall have the following elements:

- The `replyTo` element of the `DERControl` resource shall map to the available URI location where the Server accepts POST of responses from the Client devices.
- The `responseRequired` element of the `DERControl` resource shall be specified with the value of 7. See the IEEE 2030.5 standard response status code table, which requires Client devices to send DERControlResponses at each state of the `DERControl` event.

- Fixed Power Factor DERControl must be tested as specified in the CSIP specification.

Function	IEEE 2030.5 Control	Control Type
Fixed Power Factor	`opModFixedPFInjectW`	Immediate

Setting	Default	Test Values
opModFixedPFInjectW.displacement	950	900
opModFixedPFInjectW.excitation	false	false
opModFixedPFInjectW.multiplier	-3	-3

Figure 8 Fixed Power Factor Settings

Procedure

1. **[T]** Record the Client/Server communications.
2. The Client shall have retrieved all its FSA group and associated resources, including all subordinate resources for `DERProgram`, `DERControl`, and `DefaultDERControl`, `DERCurve`List and others. The Client would have processed all resource information for the FSA to find the priorities of each `DERProgram` and associated `DERControl` resources. Now, the two `DERProgram` and `DERControl` would have been scheduled, as specified in the test setup, and polling the `FSAList` for updated information, if any.
3. **[C]** Using the `DefaultDERControl` information associated with the highest priority `DERProgram`, the Client shall activate the `DefaultDERControl` until the Service Point DER event becomes active.
4. **[C]** At the effective start time of the highest priority `DERProgram/DERControl`, it shall check the `currentStatus` of the `DERControl` by executing an HTTP GET on the selected `DERControl` href. After the `currentStatus` field is verified, Client shall activate this `DERControl` using the subordinate resource information, including the immediate or curve-based control required. If a response is required, Client shall send a series of response message based on the response requirements specified in the `DERControl`.

5. **[S]** Process the sent response messages and verify they reflect the `currentStatus` of the actual DER event.
6. **[C]** Complete the current `DERControl` event at the effective end time and send the required response message to the Server. If there is no other active event, it defaults to the `DefaultDERControl` information associated with the highest priority `DERProgram`.
7. **[S]** Verify all the immediate or curve-based control is exercised in this test for the Client by checking the response message sent by the Client for each `DERControl` event.

Pass/Fail Criteria

- **[C, S]** The Client successfully detected and activated the `DefaultDERControl` specified by the Server for the highest priority `DERProgram/DERControl`.
- **[C, S]** The Client successfully polled and checked the `currentStatus` before activating the `DERControl` event. The Client successfully activated the correct inverter control mode (Fixed Power Factor Control) specified in the `DERControl`. The Client successfully sent a series of response message based on the response requirements specified in the `DERControl`.
- **[C, S]** Server successfully processed response messages sent by the Client and verified they reflect the `currentStatus` of the actual DER event.
- **[C, S]** The Client successfully completed the current `DERControl` event at the effective end time and sent the required response message to the Server.

BASIC-009 - Basic Inverter Control (Connect/Disconnect) [C, A, S]

Purpose

Verify provisioning, acquisition, and execution of connect/disconnect control.

The basic inverter control test verifies the inverter ability to process immediate and curve-based controls supported by the IEEE 2030.5 standard and specified by the CSIP specification.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in Figure 3). Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
    - `SFDI`
      Short form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the Client.
    - `LFDI`
      Long form device identifier. One of the three `EndDevice` instances must have an `LFDI` that matches the `LFDI` of the Client.
    - `RegistrationLink`
      Link to the Registration resource of the `EndDevice` instance with the `Registration:PIN` resource set to `111115`
    - `FunctionSetAssignmentsListLink`
      Link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
4. **[S]** Construct a `FunctionSetAssignmentsList` resource for the Client with two `FunctionSetAssignments` (topology and non-topology FSAs) following Figure 3. The `DERPrograms` for the topology `FunctionSetAssignment` instance for the Client shall be configured as follows:
    - Closest node/FSA to the DER device (Service Point) shall include a `DERProgram` with the highest priority (lowest primacy value).
    - At least one `DERPrograms` shall have a `DefaultDERControl` with a valid DER inverter control the DER device must activate during idle periods.
    - The Service Point `DERProgram` shall have a `DERControl` instance with fixed Connect/Disconnect values in Figure 9 Connect/Disconnect Settings.
    - The `DERControl` shall have an effective start time of (plus-two minutes) and duration (plus-one minute) specified. The `DERControl` shall have the following elements:

- The `replyTo` element of the `DERControl` resource shall map to the available URI location where the Server accepts POST of responses from the Client devices.
- The `responseRequired` element of the `DERControl` resource shall be specified with the value of 7. See the IEEE 2030.5 standard response status code table, which requires Client devices to send DERControlResponses at each state of the `DERControl` event.

- Connect/Disconnect DERControl must be tested as specified in the CSIP specification.

Function	IEEE 2030.5 Control	Control Type
Connect/Disconnect	`opModEnergize` `opModConnect` [optional]	Immediate

Setting	Default	Test Values
opModEnergize	true	true
opmodConnect		true

Figure 9 Connect/Disconnect Settings

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** The Client shall have retrieved all its FSA group and associated resources, including all subordinate resources for `DERProgram`, `DERControl`, and `DefaultDERControl`, `DERCurve`List and others. The Client would have processed all resource information for the FSA to find the priorities of each `DERProgram` and associated `DERControl` resources. Now, the two `DERProgram` and `DERControl` would have been scheduled, as specified in the test setup, and polling the `FSAList` for updated information, if any.
3. **[C]** Using the `DefaultDERControl` information associated with the highest priority `DERProgram`, the Client shall activate the `DefaultDERControl` until the Service Point DER event becomes active.
4. **[C]** At the effective start time of the highest priority `DERProgram/DERControl`, it shall check the `currentStatus` of the `DERControl` by executing an HTTP GET on the selected `DERControl` href. After the `currentStatus` field is verified, Client shall activate this `DERControl` using the subordinate resource information, including the immediate or curve-based control required. If a response is required, Client shall send a series of response message based on the response requirements specified in the `DERControl`.

5. **[S]** Process the sent response messages and verify they reflect the `currentStatus` of the actual DER event.
6. **[C]** Complete the current `DERControl` event at the effective end time and send the required response message to the Server. If there is no other active event, it defaults to the `DefaultDERControl` information associated with the highest priority `DERProgram`.
7. **[S]** Verify all the immediate or curve-based control is exercised in this test for the Client by checking the response message sent by the Client for each `DERControl` event.

Pass/Fail Criteria

- **[C, S]** The Client successfully detected and activated the `DefaultDERControl` specified by the Server for the highest priority `DERProgram/DERControl`.
- **[C, S]** The Client successfully polled and checked the `currentStatus` before activating the `DERControl` event. The Client successfully activated the correct inverter control mode (Connect/Disconnect) specified in the `DERControl`. The Client successfully sent a series of response message based on the response requirements specified in the `DERControl`.
- **[C, S]** Server successfully processed response messages sent by the Client and verified they reflect the `currentStatus` of the actual DER event.
- **[C, S]** The Client successfully completed the current `DERControl` event at the effective end time and sent the required response message to the Server.

BASIC-010 - Basic Inverter Control (Limit Max Active Power Mode) [C, A, S]

Purpose

Verify provisioning, acquisition, and execution of limit max active power.

The basic inverter control test verifies the inverter ability to process immediate and curve-based controls supported by the 2030.5 standard and specified by the CSIP specification.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in Figure 3). Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
    - `sFDI`
      Short form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the Client.
    - `RegistrationLink`
      Link to the Registration resource of the `EndDevice` instance with the `Registration:PIN` resource set to `111115`
    - `FunctionSetAssignmentsListLink`
      Link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
4. **[S]** Construct a `FunctionSetAssignmentsList` resource for the Client with two `FunctionSetAssignments` (topology and non-topology FSAs) following Figure 3. The `DERPrograms` for the topology `FunctionSetAssignment` instance for the Client shall be configured as follows:
    - Closest node/FSA to the DER device (Service Point) shall include a `DERProgram` with the highest priority (lowest primacy value).
    - At least one `DERPrograms` shall have a `DefaultDERControl` with a valid DER inverter control the DER device must activate during idle periods.
    - The Service Point `DERProgram` shall have a `DERControl` instance with Limit Max Active Power values in Figure 10 Limit Max Active Power Settings.
    - The `DERControl` shall have an effective start time of (plus-two minutes) and duration (plus-one minute) specified. The `DERControl` shall have the following elements:

- The `replyTo` element of the `DERControl` resource shall map to the available URI location where the Server accepts POST of responses from the Client devices.
- The `responseRequired` element of the `DERControl` resource shall be specified with the value of 7. See the IEEE 2030.5 standard response status code table, which requires Client devices to send DERControlResponses at each state of the `DERControl` event.

- Limit Max Active Power must be tested as specified in the CSIP specification.

Function	IEEE 2030.5 Control	Control Type
Limit Max Active Power Mode	`opModMaxLimW`	Immediate

Setting	Default	Test Values
opModMaxLimW	5000 (50%)	6000

Figure 10 Limit Max Active Power Settings

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** The Client shall have retrieved all its FSA group and associated resources, including all subordinate resources for `DERProgram`, `DERControl`, and `DefaultDERControl`, `DERCurve`List and others. The Client would have processed all resource information for the FSA to find the priorities of each `DERProgram` and associated `DERControl` resources. Now, the two `DERProgram` and `DERControl` would have been scheduled, as specified in the test setup, and polling the `FSAList` for updated information, if any.
3. **[C]** Using the `DefaultDERControl` information associated with the highest priority `DERProgram`, the Client shall activate the `DefaultDERControl` until the Service Point DER event becomes active.
4. **[C]** At the effective start time of the highest priority `DERProgram/DERControl`, it shall check the `currentStatus` of the `DERControl` by executing an HTTP GET on the selected `DERControl` href. After the `currentStatus` field is verified, Client shall activate this `DERControl` using the subordinate resource information, including the immediate or curve-based control required. If a response is required, Client shall send a series of response message based on the response requirements specified in the `DERControl`.
5. **[S]** Process the sent response messages and verify they reflect the `currentStatus` of the actual DER event.
6. **[C]** Complete the current `DERControl` event at the effective end time and send the required response message to the Server. If there is no other active event, it defaults to

the DefaultDERControl information associated with the highest priority DERProgram.

7. **[S]** Verify all the immediate or curve-based control is exercised in this test for the Client by checking the response message sent by the Client for each DERControl event.

Pass/Fail Criteria

- **[C, S]** The Client successfully detected and activated the DefaultDERControl specified by the Server for the highest priority DERProgram/DERControl.
- **[C, S]** The Client successfully polled and checked the currentStatus before activating the DERControl event. The Client successfully activated the correct inverter control mode (Limit Max Active Power) specified in the DERControl. The Client successfully sent a series of response message based on the response requirements specified in the DERControl.
- **[C, S]** Server successfully processed response messages sent by the Client and verified they reflect the currentStatus of the actual DER event.
- **[C, S]** The Client successfully completed the current DERControl event at the effective end time and sent the required response message to the Server.

BASIC-011 - Basic Inverter Control (Volt-Watt) [C, A, S]

Purpose

Verify provisioning, acquisition, and execution of volt-watt control.

The basic inverter control test verifies the inverter ability to process immediate and curve-based controls supported by the 2030.5 standard and specified by the CSIP specification.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in Figure 3). Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
    - `SFDI`
      Short form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the Client.
    - `RegistrationLink`
      Link to the Registration resource of the `EndDevice` instance with the `Registration:PIN` resource set to `111115`
    - `FunctionSetAssignmentsListLink`
      Link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
4. **[S]** Construct a `FunctionSetAssignmentsList` resource for the Client with two `FunctionSetAssignments` (topology and non-topology FSAs) following Figure 3. The `DERPrograms` for the topology `FunctionSetAssignment` instance for the Client shall be configured as follows:
    - Closest node/FSA to the DER device (Service Point) shall include a `DERProgram` with the highest priority (lowest primacy value).
    - At least one `DERPrograms` shall have a `DefaultDERControl` with a valid DER inverter control the DER device must activate during idle periods.
    - The Service Point `DERProgram` shall have a `DERControl` instance with Volt-Watt values in Figure 11 Volt-Watt Settings.
    - The `DERControl` shall have an effective start time of (plus-two minutes) and duration (plus-one minute) specified. The `DERControl` shall have the following elements:
        - The `replyTo` element of the `DERControl` resource shall map to the available URI location where the Server accepts POST of responses from the Client devices.

- The `responseRequired` element of the `DERControl` resource shall be specified with the value of 7. See the IEEE 2030.5 standard response status code table, which requires Client devices to send DERControlResponses at each state of the `DERControl` event.

- Volt-Watt DERControl must be tested as specified in the CSIP specification.

Function	IEEE 2030.5 Control	Control Type
Volt-Watt	`opModVoltWatt`	Curve

Setting	Default	Test Values
opModVoltWatt.DERCurve.CurveData	(10000,10000),(10600,10000),(11000,0)	(10000,10000),(10500,10000),(10900,0)
DERCurve.curveType	12	12
DERCurve.xMultiplier	-2	-2
DERCurve.yMultiplier	-2	-2
DERCurve.yRefType	1	1

Figure 11 Volt-Watt Settings

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** The Client shall have retrieved all its FSA group and associated resources, including all subordinate resources for `DERProgram`, `DERControl`, and `DefaultDERControl`, `DERCurve`List and others. The Client would have processed all resource information for the FSA to find the priorities of each `DERProgram` and associated `DERControl` resources. Now, the two `DERProgram` and `DERControl` would have been scheduled, as specified in the test setup, and polling the `FSAList` for updated information, if any.
3. **[C]** Using the `DefaultDERControl` information associated with the highest priority `DERProgram`, the Client shall activate the `DefaultDERControl` until the Service Point DER event becomes active.
4. **[C]** At the effective start time of the highest priority `DERProgram/DERControl`, it shall check the `currentStatus` of the `DERControl` by executing an HTTP GET on the selected `DERControl` href. After the `currentStatus` field is verified, Client shall activate this `DERControl` using the subordinate resource information, including the immediate or curve-based control required. If a response is required, Client shall send a series of response message based on the response requirements specified in the `DERControl`.
5. **[S]** Process the sent response messages and verify they reflect the `currentStatus` of the actual DER event.

6. **[C]** Complete the current `DERControl` event at the effective end time and send the required response message to the Server. If there is no other active event, it defaults to the `DefaultDERControl` information associated with the highest priority `DERProgram`.
7. **[S]** Verify all the immediate or curve-based control is exercised in this test for the Client by checking the response message sent by the Client for each `DERControl` event.

Pass/Fail Criteria

- **[C, S]** The Client successfully detected and activated the `DefaultDERControl` specified by the Server for the highest priority `DERProgram/DERControl`.
- **[C, S]** The Client successfully polled and checked the `currentStatus` before activating the `DERControl` event. The Client successfully activated the correct inverter control mode (Volt-Watt) specified in the `DERControl`. The Client successfully sent a series of response message based on the response requirements specified in the `DERControl`.
- **[C, S]** Server successfully processed response messages sent by the Client and verified they reflect the `currentStatus` of the actual DER event.
- **[C, S]** The Client successfully completed the current `DERControl` event at the effective end time and sent the required response message to the Server.

BASIC-012 - Basic Inverter Control (Frequency-Watt) [C, A, S]

Purpose

Verify provisioning, acquisition, and execution of frequency-watt control

The basic inverter control test verifies the inverter ability to process immediate and curve-based controls supported by the 2030.5 standard and specified by the CSIP specification.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in Figure 3). Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
    - `SFDI`
      Short form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the Client.
    - `RegistrationLink`
      Link to the Registration resource of the `EndDevice` instance with the `Registration:PIN` resource set to `111115`
    - `FunctionSetAssignmentsListLink`
      Link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
4. **[S]** Construct a `FunctionSetAssignmentsList` resource for the Client with two `FunctionSetAssignments` (topology and non-topology FSAs) following Figure 3. The `DERPrograms` for the topology `FunctionSetAssignment` instance for the Client shall be configured as follows:
    - Closest node/FSA to the DER device (Service Point) shall include a `DERProgram` with the highest priority (lowest primacy value).
    - At least one `DERPrograms` shall have a `DefaultDERControl` with a valid DER inverter control the DER device must activate during idle periods.
    - The Service Point `DERProgram` shall have a `DERControl` instance with Frequency-Watt values in Figure 12 Frequency-Watt Settings.
    - The `DERControl` shall have an effective start time of (plus-two minutes) and duration (plus-one minute) specified. The `DERControl` shall have the following elements:
        - The `replyTo` element of the `DERControl` resource shall map to the available URI location where the Server accepts POST of responses from the Client devices.

- The `responseRequired` element of the `DERControl` resource shall be specified with the value of 7. See the IEEE 2030.5 standard response status code table, which requires Client devices to send DERControlResponses at each state of the `DERControl` event.

- Frequency-Watt DERControl must be tested as specified in the CSIP specification.

Function	IEEE 2030.5 Control	Control Type
Frequency-Watt	`opModFreqWatt`	Curve

Setting	Default	Test Values
opModFreqDroop.dBOF	60036 (60.036)	60030
opModFreqDroop.dBUF	59964 (59.964)	59970
opModFreqDroop.kOF	50 (.05)	40
opModFreqDroop.kUF	50 (.05)	40
opModFreqDroop.openLoopTms	500 (5 sec)	600
opModFreqWatt.DERCurve.CurveData		(5900,100), (5950,80), (6050,80), (6200, 0)
DERCurve.curveType		0
DERCurve.xMultiplier		-2
DERCurve.yMultiplier		0
DERCurve.yRefType		1

Figure 12 Frequency-Watt Settings

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** The Client shall have retrieved all its FSA group and associated resources, including all subordinate resources for `DERProgram`, `DERControl`, and `DefaultDERControl`, `DERCurve`List and others. The Client would have processed all resource information for the FSA to find the priorities of each `DERProgram` and associated `DERControl` resources. Now, the two `DERProgram` and `DERControl` would have been scheduled, as specified in the test setup, and polling the `FSAList` for updated information, if any.
3. **[C]** Using the `DefaultDERControl` information associated with the highest priority `DERProgram`, the Client shall activate the `DefaultDERControl` until the Service Point DER event becomes active.

4. **[C]** At the effective start time of the highest priority `DERProgram/DERControl`, it shall check the `currentStatus` of the `DERControl` by executing an HTTP GET on the selected `DERControl` href. After the `currentStatus` field is verified, Client shall activate this `DERControl` using the subordinate resource information, including the immediate or curve-based control required. If a response is required, Client shall send a series of response message based on the response requirements specified in the `DERControl`.
5. **[S]** Process the sent response messages and verify they reflect the `currentStatus` of the actual DER event.
6. **[C]** Complete the current `DERControl` event at the effective end time and send the required response message to the Server. If there is no other active event, it defaults to the `DefaultDERControl` information associated with the highest priority `DERProgram`.
7. **[S]** Verify all the immediate or curve-based control is exercised in this test for the Client by checking the response message sent by the Client for each `DERControl` event.

Pass/Fail Criteria

- **[C, S]** The Client successfully detected and activated the `DefaultDERControl` specified by the Server for the highest priority `DERProgram/DERControl`.
- **[C, S]** The Client successfully polled and checked the `currentStatus` before activating the `DERControl` event. The Client successfully activated the correct inverter control mode (Frequency-Watt) specified in the `DERControl`. The Client successfully sent a series of response message based on the response requirements specified in the `DERControl`.
- **[C, S]** Server successfully processed response messages sent by the Client and verified they reflect the `currentStatus` of the actual DER event.
- **[C, S]** The Client successfully completed the current `DERControl` event at the effective end time and sent the required response message to the Server.

BASIC-013 - Basic Inverter Control (Set Active Power Mode - in % of Max Power)

Purpose

Verify provisioning, acquisition, and execution of set active power control.

The basic inverter control test verifies the inverter ability to process immediate and curve-based controls supported by the 2030.5 standard and specified by the CSIP specification.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in Figure 3). Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
    - `sFDI`
      Short form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the Client.
    - `RegistrationLink`
      Link to the Registration resource of the `EndDevice` instance with the `Registration:PIN` resource set to `111115`
    - `FunctionSetAssignmentsListLink`
      Link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
4. **[S]** Construct a `FunctionSetAssignmentsList` resource for the Client with two `FunctionSetAssignments` (topology and non-topology FSAs) following Figure 3. The `DERPrograms` for the topology `FunctionSetAssignment` instance for the Client shall be configured as follows:
    - Closest node/FSA to the DER device (Service Point) shall include a `DERProgram` with the highest priority (lowest primacy value).
    - At least one `DERPrograms` shall have a `DefaultDERControl` with a valid DER inverter control the DER device must activate during idle periods.
    - The Service Point `DERProgram` shall have a `DERControl` instance with Set Active Power Mode values in Figure 13 Set Active Power (% Max Power) Settings.
    - The `DERControl` shall have an effective start time of (plus-two minutes) and duration (plus-one minute) specified. The `DERControl` shall have the following elements:

- The `replyTo` element of the `DERControl` resource shall map to the available URI location where the Server accepts POST of responses from the Client devices.
- The `responseRequired` element of the `DERControl` resource shall be specified with the value of 7. See the IEEE 2030.5 standard response status code table, which requires Client devices to send DERControlResponses at each state of the `DERControl` event.

- Set Active Power Mode DERControl must be tested as specified in the CSIP specification.

Function	IEEE 2030.5 Control	Control Type
Set Active Power Mode (% of Max Power)	`opModFixedW`	Immediate

Setting	Default	Test Values
opModFixedW	5000 (50%)	6000

Figure 13 Set Active Power (% Max Power) Settings

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** The Client shall have retrieved all its FSA group and associated resources, including all subordinate resources for `DERProgram`, `DERControl`, and `DefaultDERControl`, `DERCurve`List and others. The Client would have processed all resource information for the FSA to find the priorities of each `DERProgram` and associated `DERControl` resources. Now, the two `DERProgram` and `DERControl` would have been scheduled, as specified in the test setup, and polling the `FSAList` for updated information, if any.
3. **[C]** Using the `DefaultDERControl` information associated with the highest priority `DERProgram`, the Client shall activate the `DefaultDERControl` until the Service Point DER event becomes active.
4. **[C]** At the effective start time of the highest priority `DERProgram/DERControl`, it shall check the `currentStatus` of the `DERControl` by executing an HTTP GET on the selected `DERControl` href. After the `currentStatus` field is verified, Client shall activate this `DERControl` using the subordinate resource information, including the immediate or curve-based control required. If a response is required, Client shall send a series of response message based on the response requirements specified in the `DERControl`.
5. **[S]** Process the sent response messages and verify they reflect the `currentStatus` of the actual DER event.

6. **[C]** Complete the current `DERControl` event at the effective end time and send the required response message to the Server. If there is no other active event, it defaults to the `DefaultDERControl` information associated with the highest priority `DERProgram`.
7. **[S]** Verify all the immediate or curve-based control is exercised in this test for the Client by checking the response message sent by the Client for each `DERControl` event.

Pass/Fail Criteria

- **[C, S]** The Client successfully detected and activated the `DefaultDERControl` specified by the Server for the highest priority `DERProgram/DERControl`.
- **[C, S]** The Client successfully polled and checked the `currentStatus` before activating the `DERControl` event. The Client successfully activated the correct inverter control mode (Set Active Power Mode) specified in the `DERControl`. The Client successfully sent a series of response message based on the response requirements specified in the `DERControl`.
- **[C, S]** Server successfully processed response messages sent by the Client and verified they reflect the `currentStatus` of the actual DER event.
- **[C, S]** The Client successfully completed the current `DERControl` event at the effective end time and sent the required response message to the Server.

BASIC-014 - Basic Inverter Control (Set Active Power Mode – in Watts)

Purpose

Verify provisioning, acquisition, and execution of set active power control.

The basic inverter control test verifies the inverter ability to process immediate and curve-based controls supported by the 2030.5 standard and specified by the CSIP specification.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in Figure 3). Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
    - `SFDI`
      Short form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the Client.
    - `RegistrationLink`
      Link to the Registration resource of the `EndDevice` instance with the `Registration:PIN` resource set to `111115`
    - `FunctionSetAssignmentsListLink`
      Link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
4. **[S]** Construct a `FunctionSetAssignmentsList` resource for the Client with two `FunctionSetAssignments` (topology and non-topology FSAs) following Figure 3. The `DERPrograms` for the topology `FunctionSetAssignment` instance for the Client shall be configured as follows:
    - Closest node/FSA to the DER device (Service Point) shall include a `DERProgram` with the highest priority (lowest primacy value).
    - At least one `DERPrograms` shall have a `DefaultDERControl` with a valid DER inverter control the DER device must activate during idle periods.
    - The Service Point `DERProgram` shall have a `DERControl` instance with Set Active Power Mode values in Figure 14 Set Active Power (Watts) Settings.
    - The `DERControl` shall have an effective start time of (plus-two minutes) and duration (plus-one minute) specified. The `DERControl` shall have the following elements:
        - The `replyTo` element of the `DERControl` resource shall map to the available URI location where the Server accepts POST of responses from the Client devices.

- The responseRequired element of the DERControl resource shall be specified with the value of 7. See the IEEE 2030.5 standard response status code table, which requires Client devices to send DERControlResponses at each state of the DERControl event.

- Set Active Power Mode DERControl must be tested as specified in the CSIP specification.

Function	IEEE 2030.5 Control	Control Type
Set Active Power Mode (in Watts)	opModTargetW	Immediate

Setting	Default	Test Values
opModTargetW	2000 (value 2000, multiplier 0)	3000 (0 multiplier)

Figure 14 Set Active Power (Watts) Settings

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** The Client shall have retrieved all its FSA group and associated resources, including all subordinate resources for DERProgram, DERControl, and DefaultDERControl, DERCurveList and others. The Client would have processed all resource information for the FSA to find the priorities of each DERProgram and associated DERControl resources. Now, the two DERProgram and DERControl would have been scheduled, as specified in the test setup, and polling the FSAList for updated information, if any.
3. **[C]** Using the DefaultDERControl information associated with the highest priority DERProgram, the Client shall activate the DefaultDERControl until the Service Point DER event becomes active.
4. **[C]** At the effective start time of the highest priority DERProgram/DERControl, it shall check the currentStatus of the DERControl by executing an HTTP GET on the selected DERControl href. After the currentStatus field is verified, Client shall activate this DERControl using the subordinate resource information, including the immediate or curve-based control required. If a response is required, Client shall send a series of response message based on the response requirements specified in the DERControl.
5. **[S]** Process the sent response messages and verify they reflect the currentStatus of the actual DER event.
6. **[C]** Complete the current DERControl event at the effective end time and send the required response message to the Server. If there is no other active event, it defaults to the DefaultDERControl information associated with the highest priority DERProgram.

7. **[S]** Verify all the immediate or curve-based control is exercised in this test for the Client by checking the response message sent by the Client for each `DERControl` event.

Pass/Fail Criteria

- **[C, S]** The Client successfully detected and activated the `DefaultDERControl` specified by the Server for the highest priority `DERProgram/DERControl`.
- **[C, S]** The Client successfully polled and checked the `currentStatus` before activating the `DERControl` event. The Client successfully activated the correct inverter control mode (Set Active Power Mode) specified in the `DERControl`. The Client successfully sent a series of response message based on the response requirements specified in the `DERControl`.
- **[C, S]** Server successfully processed response messages sent by the Client and verified they reflect the `currentStatus` of the actual DER event.
- **[C, S]** The Client successfully completed the current `DERControl` event at the effective end time and sent the required response message to the Server.

BASIC-015 – Advanced Inverter Control [C, A, S]

Purpose

The advanced inverter control test verifies the inverter ability to process the advanced inverter control conditions and event scenarios.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Create an `EndDeviceList` with three `EndDevice` instances, with one `EndDevice` instance representing the Client, Inverter-A in the above diagram). Make sure the changedTime of the `EndDevice` instances have a more recent time than the Client instance. This ensures the Client instance is the last in the list.
3. **[S]** For each `EndDevice`, these fields must be populated:
   - `SFDI`
     Short form device identifier. One of the three `EndDevice` instances must have an `SFDI` that matches the `SFDI` of the Client.
   - `LFDI`
     Long form device identifier. One of the three `EndDevice` instances must have an `LFDI` that matches the `LFDI` of the Client.
   - `RegistrationLink`
     Link to the Registration resource of the `EndDevice` instance with the `Registration:PIN` resource set to `111115`
   - `FunctionSetAssignmentsListLink`
     Link to the `FunctionSetAssignmentsList` resource of the `EndDevice` instance.
4. **[S]** Construct a `FunctionSetAssignmentsList` resource for the Client with two `FunctionSetAssignments` (topology and non-topology FSAs) following the above example diagram. The `DERPrograms` for the topology `FunctionSetAssignment` instance for the Client shall be configured as follows:
   - Closest node/FSA to the DER device (Service Point) shall include a `DERProgram` with the highest priority (lowest primacy value). Next closest node/FSA to the DER device (Transformer) shall include a `DERProgram` with second-highest priority (second-lowest primacy value).
   - At least one `DERPrograms` shall have a `DefaultDERControl` with a valid DER inverter control the DER device must activate during idle periods.
   - The Service Point `DERProgram` shall have twenty-four `DERControl` instances with Fixed Power Factor opMod* function. `DERControl#N` shall have an effective start time of (plus-two *N minutes) and duration (plus-one minute) specified.

For example, DERControl#1 starts at plus-two minutes with duration of plus-one minute. DERControl#2 starts at plus-four minutes with duration of plus-one minute, etc.

- The Transformer DERProgram shall have one DERControl instance(s) with Fixed Power Factor opMod* function. DERControl#1 shall have an effective start time of (plus-two minutes) and duration (plus-one minute) specified.
- Both DERControls shall have the following elements:
    - The replyTo element of the DERControl resource shall map to the available URI location where the Server accepts POST of responses from the Client devices.
    - The responseRequired element of the DERControl resource shall be specified with the value of 7. See the IEEE 2030.5 standard response status code table, which requires Client devices to send DERControlResponses at each state of the DERControl event.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** The Client shall have retrieved all its FSA group and associated resources, including all subordinate resources for DERProgram, DERControl, and DefaultDERControl, DERCurveList and others. The Client would have processed all resource information for the FSA to find the priorities of each DERProgram and associated DERControl resources. Now, all the DERProgram and DERControl would have been scheduled in successive fashion, as specified in the test setup, and polling the FSAList for updated information, if any.
3. **[C]** Using the DefaultDERControl information associated with the highest priority DERProgram, the Client shall activate the DefaultDERControl until the third DER event becomes active.
4. **[C]** At the effective start time of the first DERProgram/DERControl, it shall check the currentStatus of the DERControl by executing an HTTP GET on the selected DERControl href. After currentStatus field is verified, Client shall activate this DERControl (Fixed Power Factor) using the subordinate resource information, including the immediate or curve-based control required. If a response is required, Client shall send a series of response message based on the response requirements specified in the DERControl.
5. **[S]** Process the sent response messages and verify they reflect the currentStatus of the actual DER event.
6. **[C]** Complete the current DERControl event at the effective end time and send the required response message to the Server.

7. **[S]** Cancel the remaining highest priority `DERProgram/DERControl` instances by updating `currentStatus` to `2 = Canceled`, to indicate cancellation.
8. **[C]** By checking the `currentStatus` included in step 6, Client shall recognize the remaining `DERProgram/DERControl` instances have now been canceled. Client shall ignore this event and move onto the next event, if applicable, or revert back to the correct DefaultDERControl.
9. **[S]** Verify all the immediate or curve-based control is exercised in this test for the Client by checking the response message sent by the Client for each `DERControl` event.

Pass/Fail Criteria

- **[C, S]** The Client successfully detected and activated the `DefaultDERControl` specified by the Server for the third (highest priority) `DERProgram/DERControl`.
- **[C, S]** The Client successfully checked the `currentStatus` before activating the `DERControl` event. The Client successfully activated the correct inverter control mode (Fixed Power Factor) specified in the `DERControl`. The Client successfully sent a series of response message based on the response requirements specified in the `DERControl`.
- **[C, S]** Server successfully processed response messages sent by the Client and verified they reflect the `currentStatus` of the actual DER event.
- **[C, S]** The Client successfully completed the current `DERControl` event at the effective end time and sent the required response message to the Server.
- **[S]** Server successfully canceled the remaining `DERProgram/DERControl` event before it started by updating the `currentStatus` to `2 = Canceled`.
- **[C, S]** Client, by checking the remaining events' `currentStatus`, recognized the remaining `DERProgram/DERControl`s were canceled. The Client successfully detected that there are no other events scheduled after the first one and reverts to using Service Point `DERProgram DefaultDERControl`.

BASIC-016 - Event - 2 DERP, 2 DDERC, 0 DERC [C, A, S]

Purpose

The event (2 DERP, 2 DDERC, 0 DERC) test verifies that the Client executes the correct default DER control from one of its `DERPrograms`, or groups.

Setup

1. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`.
2. **[S]** Create a `DefaultDERControl` for the same specific control (for example, `opModFixedPFInjectW`) in the SP `DERProgram`.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** GETs the SY `DERProgramList`.
3. **[C]** GETs the SY `DERProgram`.
4. **[C]** GETs the SY `DefaultDERControl`.
5. **[C]** GETs the SP `DERProgramList`.
6. **[C]** GETs the SP `DERProgram`.
7. **[C]** GETs the SP `DefaultDERControl`.
8. **[C]** Applies the SP `DefaultDERControl` because it has a higher priority (lower primacy value) than the SY `DefaultDERControl`.

Pass/Fail Criteria

- **[C, S]** Client GETs SY `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DefaultDERControl` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DefaultDERControl` with a response code of `200 OK`.
- **[C]** Client applies the SP `DefaultDERControl` because it has the higher priority (lower primacy value). Note that there is no acknowledgment for activating a `DefaultDERControl` in the IEEE 2030.5 protocol. Verification of `DefaultDERControl` activation must be done out-of-band.

BASIC-017 - Event - 1 DERP, 0 DDERC, 1 DERC [C, A, S]

Purpose

The event (DERP, 0 DDERC, 1 DERC) test verifies that the Client executes a single DER event from one of its `DERPrograms`, or groups.

Setup

1. **[S]** Create a `DERControl` (for example, `opModFixedPFInjectW`) with a start time of two minutes from now and duration of one minute.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** GETs the SY `DERProgramList`.
3. **[C]** GETs the SY `DERProgram`.
4. **[C]** Periodically GETs the SY `DERControlList`.
5. **[C]** GETs of the created SY `DERControl`.
6. **[C]** POSTs response with status 1 (Event Received) to the Server.
7. **[C]** Applies the SY `DERControl` at the correct start time for the correct duration.
8. **[C]** POSTs response with status 2 (Event Started) at start time of the event.
9. **[C]** POSTs response with status 3 (Event Completed) after duration has elapsed.

Pass/Fail Criteria

- **[C, S]** Client GETs SY `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs the created SY `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client applies the SY `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client POSTs response with status 2 (Event Started) at start time. Server receives the responses and sends a response code of `201 Created`.
- **[C, S]** Client POSTs response with status 3 (Event Completed) after duration has elapsed. Server receives the responses and sends a response code of `201 Created`.

BASIC-018 - Event - 1 DERP, 1 DDERC, 1 DERC [C, A, S]

Purpose

The event (1 DERP, 1 DDERC, 1 DERC) test verifies that the Client executes a single DER event from one of its `DERPrograms` and applies the `DefaultDERC` when the DER event is not active.

Setup

1. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`.
2. **[S]** Create a `DERControl` (for example, `opModFixedPFInjectW`) with a start time of two minutes from now and duration of one minute.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** GETs the SY `DERProgramList`.
3. **[C]** GETs the SY `DERProgram`.
4. **[C]** GETs the SY `DefaultDERControl`.
5. **[C]** Applies the SY `DefaultDERControl` because there is no active event.
6. **[C]** Periodically GETs the SY `DERControlList`.
7. **[C]** GETs the created SY `DERControl`.
8. **[C]** POSTs response with status 1 (Event Received) to the Server.
9. **[C]** Applies the SY `DERControl` at the correct start time for the correct duration.
10. **[C]** POSTs response with status 2 (Event Started) at start time.
11. **[C]** POSTs response with status 3 (Event Completed) after duration has elapsed.
12. **[C]** Applies the SY `DefaultDERControl` after the completion of the event.

Pass/Fail Criteria

- **[C, S]** Client GETs SY `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DefaultDERControl` with a response code of `200 OK`.
- **[C, S]** Client applies the SY `DefaultDERControl` because there are no events active.
- **[C, S]** Client GETs SY `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs the created SY `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client applies the System FSA `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client POSTs response with status 2 (Event Started) at start time. Server receives the responses and sends a response code of `201 Created`.
- Client POSTs response with status 3 (Event Completed) after duration has elapsed.

- Server receives the responses and sends a response code of `201 Created`.
- Client applies the SY `DefaultDERControl` after the event has completed.

BASIC-019 - Event - 1 DERP, 1 DDERC, 2 Non-overlapping Similar DERC [C, A, S]

Purpose

The event (DERP, 1 DDERC, 2 non-overlapping similar DERC) test verifies that the Client executes two similar, non-overlapping DER events with a `DefaultDERC` from one of its `DERPrograms`.

Setup

1. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the System `DERProgram`.
2. **[S]** Create a `DERControl` (for example, `opModFixedPFInjectW`) with a start time of two minutes from now and duration of one minute.
3. **[S]** Create a `DERControl` for the same specific control with a start time of four minutes from now duration of one minute.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** GETs the SY `DERProgramList`.
3. **[C]** GETs the SY `DERProgram`.
4. **[C]** GETs the SY `DefaultDERControl`.
5. **[C]** Applies the SY `DefaultDERControl` because there is no active event.
6. **[C]** Periodically GETs the SY `DERControlList`.
7. **[C]** GETs of the SY `DERControl` event created in setup step 2.
8. **[C]** POSTs response with status 1 (Event Received) for the event created in setup step 2 to the Server.
9. **[C]** Applies the SY `DERControl` created in Setup Step 2 at the correct start time for the correct duration.
10. **[C]** POSTs response with status 2 (Event Started) at start time of event created in setup step 2.
11. **[C]** POSTs response with status 3 (Event Completed) after duration of the event created in setup step 2 has elapsed.
12. **[C]** Applies the SY `DefaultDERControl` after the completion of the event created in setup step 2.
13. **[C]** GETs of the SY `DERControl` event created in setup step 3.
14. **[C]** POSTs response with status 1 (Event Received) for the event created in setup step 3 to the Server.
15. **[C]** Applies the SY `DERControl` created in setup step 3 at the correct start time for the correct duration.
16. **[C]** POSTs response with status 2 (Event Started) at start time of the event created in setup step 3.

17. **[C]** POSTs response with status 3 (Event Completed) after duration of the event created in setup step 3 has elapsed.
18. **[C]** Applies the SY `DefaultDERControl` after the completion of the event created in setup step 3.

Pass/Fail Criteria

- **[C, S]** Client GETs SY `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DefaultDERControl` with a response code of `200 OK`.
- **[C]** Client applies the System FSA `DefaultDERControl` because there are no events active.
- **[C, S]** Client GETs SY `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs the third SY `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the third event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client applies the third SY `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client POSTs response with status 2 (Event Started) at start time of the third event. Server receives the responses and sends a response code of `201 Created`.
- **[C, S]** Client POSTs response with status 3 (Event Completed) after duration of the third event has elapsed. Server receives the responses and sends a response code of `201 Created`.
- **[C]** Client applies the SY `DefaultDERControl` after the third event has completed.
- **[C, S]** Client GETs the second SY `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the second event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client applies the second SY `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client POSTs response with status 2 (Event Started) at start time of the second event. Server receives the responses and sends a response code of `201 Created`.
- **[C, S]** Client POSTs response with status 3 (Event Completed) after duration of the second event has elapsed. Server receives the responses and sends a response code of `201 Created`.
- **[C]** Client applies the SY `DefaultDERControl` after the second event has completed.

BASIC-020 - Event - 2 DERP, 2 DDERC, 2 Non-overlapping Similar DERC [C, A, S]

Purpose

The event (2 DERP, 2 DDERC, 2 non-overlapping similar DERC) test verifies that the Client executes two similar, non-overlapping DER events with a `DefaultDERC` from two `DERPrograms`. The two `DERPrograms` have different primacy, so the Client must use the `DefaultDERC` from the program with the lower primacy value (higher priority).

Setup

1. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`.
2. **[S]** Create a SY `DERControl` (for example, `opModFixedPFInjectW`) with a start time of two minutes from now and duration of one minute.
3. **[S]** Create a SP `DefaultDERControl` for the specific control (for example, `opModFixedPFInjectW`) for the SP `DERProgram`.
4. **[S]** Create a SP `DERControl` for the same specific control with a start time of four minutes from now and duration of one minute.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** GETs the SY `DERProgramList`.
3. **[C]** GETs the SY `DERProgram`.
4. **[C]** GETs the SY `DefaultDERControl`.
5. **[C]** GETs the SP `DERProgramList`.
6. **[C]** GETs the SP `DERProgram`.
7. **[C]** GETs the SP `DefaultDERControl`.
8. **[C]** Applies the SP `DefaultDERControl` because there is no active event and has a higher priority than the SY `DefaultDERControl`.
9. **[C]** Periodically GETs the SY `DERControlList`.
10. **[C]** Periodically GETs the SP `DERControlList`.
11. **[C]** GETs of the created SY `DERControl`.
12. **[C]** POSTs response with status 1 (Event Received) to the Server.
13. **[C]** Applies the SY `DERControl` at the correct start time for the correct duration.
14. **[C]** POSTs response with status 2 (Event Started) for the SY event at start time.
15. **[C]** POSTs response with status 3 (Event Completed) for the SY event after duration has elapsed.
16. **[C]** Applies the SP `DefaultDERControl` after the completion of the event.
17. **[C]** GETs of the SP `DERControl`.
18. **[C]** POSTs response with status 1 (Event Received) to the Server.
19. **[C]** Applies the SP `DERControl` at the correct start time for the correct duration.

20. **[C]** POSTs response with status 2 (Event Started) for the SP event at start time.
21. **[C]** POSTs response with status 3 (Event Completed) for the SP event after duration has elapsed.
22. **[C]** Applies the SP `DefaultDERControl` after the completion of the event because it has a higher priority than the SY `DefaultDERControl`.

Pass/Fail Criteria

- **[C, S]** Client GETs SY `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DefaultDERControl` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DefaultDERControl` with a response code of `200 OK`.
- **[C]** Client applies the SP `DefaultDERControl` because there are no events active and it has a higher priority than the SY `DefaultDERControl`.
- **[C, S]** Client GETs SY `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs the SY FSA `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SY event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client applies the SY `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client POSTs response with status 2 (Event Started) for the SY event at start time. Server receives the responses and sends a response code of `201 Created`.
- **[C, S]** Client POSTs response with status 3 (Event Completed) for the SY event after duration has elapsed. Server receives the responses and sends a response code of `201 Created`.
- **[C]** Client applies the SP `DefaultDERControl` after the event has completed.
- **[C, S]** Client GETs the SP `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SP event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client applies the SP `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client POSTs response with status 2 (Event Started) for the SP event at start time. Server receives the responses and sends a response code of `201 Created`.
- **[C, S]** Client POSTs response with status 3 (Event Completed) for the SP event after duration has elapsed. Server receives the responses and sends a response code of `201 Created`.
- **[C]** Client applies the SP `DefaultDERControl` after the event has completed because it has a higher priority than the System FSA `DefaultDERControl`.

BASIC-021 - Event - 2 DERP, 2 DDERC, 2 Overlapping Similar DERC - System DERC followed by Service Point DERC before Start of System DERC [C, A, S]

Purpose

The event (2 DERP, 2 DDERC, 2 overlapping similar DERC - system DERC followed by service point DERC before start of system DERC) test verifies the event handling of two overlapping events using the same DER control.

The higher priority service point control overlaps with the previously scheduled system control. Both events are scheduled ahead of time. Because the service point control is higher priority and the Client should have discovered both events before the start of the events, the Client should execute only the service point event.

Setup

1. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`
2. **[S]** Create a SY `DERControl` (for example, `opModFixedPFInjectW`) with a start time of two minutes from now and duration of two minute.
3. **[S]** Create a `DefaultDERControl` for the specific control (for example, `opModFixedPFInjectW`) for the SP `DERProgram`.
4. **[S]** Create a SP `DERControl` for the same specific control with a start time of three minutes from now and duration of one minute.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** GETs the SY `DERProgramList`.
3. **[C]** GETs the SY `DERProgram`.
4. **[C]** GETs the SY `DefaultDERControl`.
5. **[C]** GETs the SP `DERProgramList`.
6. **[C]** GETs the SP `DERProgram`.
7. **[C]** GETs the SP `DefaultDERControl`.
8. **[C]** Applies the SP `DefaultDERControl` because there is no active event and has a higher priority than the SY `DefaultDERControl`.
9. **[C]** Periodically GETs the SY `DERControlList`.
10. **[C]** Periodically GETs the SP `DERControlList`.
11. **[C]** GETs the SY `DERControl`.
12. **[C]** POSTs response with status 1 (Event Received) to the Server for the SY `DERControl`.
13. **[C]** GETs the SP `DERControl`.
14. **[C]** POSTs response with status 1 (Event Received) to the Server for the SP `DERControl`.

15. **[C]** POSTs response with status 7 (Event Superseded) to the Server for the SY `DERControl`.
16. **[C]** Does not execute them SY `DERControl` because it is superseded.
17. **[C]** Applies the SP `DERControl` at the correct start time for the correct duration.
18. **[C]** POSTs response with status 2 (Event Started) at start time of the SP event.
19. **[C]** POSTs response with status 3 (Event Completed) after duration has elapsed of the SP event.
20. **[C]** Applies the SP `DefaultDERControl` after the completion of the event because it has a higher priority than the SY `DefaultDERControl`.

Pass/Fail Criteria

- **[C, S]** Client GETs SY `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DefaultDERControl` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DefaultDERControl` with a response code of `200 OK`.
- **[C]** Client applies the SP `DefaultDERControl` because there are no events active and it has a higher priority than the SY `DefaultDERControl`.
- **[C, S]** Client GETs SY `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs the SY `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SY event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C, S]** Client GETs the SP `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SP event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C, S]** Client POSTs response with status 7 (Event Superseded) for the SY event to the Server. Receives the response and sends a response code of `201 Created`.
- **[C]** Client Does not execute the SY `DERControl` because it is superseded.
- **[C]** Client Applies the SP `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client POSTs response with status 2 (Event Started) for the SP event at start time. Server receives the responses and sends a response code of `201 Created`.
- **[C, S]** Client POSTs response with status 3 (Event Completed) for the SP event after duration has elapsed. Server receives the responses and sends a response code of `201 Created`.
- **[C]** Client applies the SP `DefaultDERControl` after the event has completed because it has a higher priority than the SY `DefaultDERControl`.

BASIC-022 - Event - 2 DERP, 2 DDERC, 2 Overlapping Similar DERC - Service Point DERC followed by System DERC [C, A, S]

Purpose

The event (2 DERP, 2 DDERC, 2 overlapping similar DERC - service point DERC followed by system DERC) test verifies the event handling of two overlapping events using the same DER control.

The lower priority system control overlaps with the previously scheduled service point control. Both events are scheduled ahead of time. Because the service point control is higher priority and the Client should have discovered both events before the start of the events, the Client should execute only the service point event.

Setup

1. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`.
2. **[S]** Create a SY `DERControl` (for example, `opModFixedPFInjectW`) with a start time of three minutes from now and duration of two minute.
3. **[S]** Create a `DefaultDERControl` for the specific control (for example, `opModFixedPFInjectW`) for the SP `DERProgram`.
4. **[S]** Create a SP `DERControl` for the same specific control with a start time of two minutes from now with duration of two minute.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** GETs the SY `DERProgramList`.
3. **[C]** GETs the SY `DERProgram`.
4. **[C]** GETs the SY `DefaultDERControl`.
5. **[C]** GETs the SP `DERProgramList`.
6. **[C]** GETs the SP `DERProgram`.
7. **[C]** GETs the SP `DefaultDERControl`.
8. **[C]** Applies the SP `DefaultDERControl` because there is no active event and has a higher priority than the SY `DefaultDERControl`.
9. **[C]** Periodically GETs the SY `DERControlList`.
10. **[C]** Periodically GETs the SP FSA `DERControlList`.
11. **[C]** GETs the SY FSA `DERControl`.
12. **[C]** POSTs response with status 1 (Event Received) to the Server for the SY `DERControl`.
13. **[C]** GETs the created SP FSA `DERControl`.
14. **[C]** POSTs response with status 1 (Event Received) to the Server for the SP `DERControl`.

15. **[C]** POSTs response with status 7 (Event Superseded) to the Server for the SY `DERControl`.
16. **[C]** Does not execute them SY `DERControl` because it is superseded.
17. **[C]** Applies the SP `DERControl` at the correct start time for the correct duration.
18. **[C]** POSTs response with status 2 (Event Started) for the SP event at start time.
19. Client: POSTs response with status 3 (Event Completed) for the SP event after duration has elapsed.
20. **[C]** Applies the SP `DefaultDERControl` after the completion of the event because it has a higher priority than the SY `DefaultDERControl`.

Pass/Fail Criteria

- **[C, S]** Client GETs SY `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DefaultDERControl` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DefaultDERControl` with a response code of `200 OK`.
- **[C]** Client applies the SP `DefaultDERControl` because there are no events active and it has a higher priority than the SY `DefaultDERControl`.
- **[C, S]** Client GETs SY `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs the SY `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SY event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C, S]** Client GETs the SP `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SP event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C, S]** Client POSTs response with status 7 (Event Superseded) for the SY event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client does not execute the SY `DERControl` because it is superseded.
- **[C]** Client applies the SP `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client POSTs response with status 2 (Event Started) for the SP event at start time. Server receives the responses and sends a response code of `201 Created`.
- **[C, S]** Client POSTs response with status 3 (Event Completed) for the SP event after duration has elapsed. Server receives the responses and sends a response code of `201 Created`.
- **[C]** Client applies the SP `DefaultDERControl` after the event has completed because it has a higher priority than the SY `DefaultDERControl`.

BASIC-023 - Event - 2 DERP, 2 DDERC, 2 Overlapping Similar DERC - System DERC followed by Service Point DERC after Start of System Event [C, A, S]

Purpose

The event (2 DERP, 2 DDERC, 2 overlapping similar DERC – system DERC followed by service point DERC after start of system event) test verifies the event handling of two overlapping events using the same DER control.

The higher priority service point control overlaps with the previously scheduled system control and the service point control is scheduled after the start of the system control. Because the service point control is higher priority, the Client should execute the system control until the start time of the service point control followed by service point control execution.

Setup

1. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`.
2. **[S]** Create a SY `DERControl` (for example, `opModFixedPFInjectW`) with a start time of one minute from now and duration of four minutes.
3. **[S]** Create a `DefaultDERControl` for the specific control (for example, `opModFixedPFInjectW`) for the SP `DERProgram`.
4. **[S]** After the SY `DERControl` has started, create a SP `DERControl` for the same specific control with a start time of two minutes from now and duration of two minutes.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** GETs the SY `DERProgramList`.
3. **[C]** GETs the SY `DERProgram`.
4. **[C]** GETs the SY `DefaultDERControl`.
5. **[C]** GETs the SP `DERProgramList`.
6. **[C]** GETs the SP `DERProgram`.
7. **[C]** GETs the SP `DefaultDERControl`.
8. **[C]** Applies the SP `DefaultDERControl` because there is no active event and has a higher priority than the SY `DefaultDERControl`.
9. **[C]** Periodically GETs the SY `DERControlList`.
10. **[C]** GETs the created SY `DERControl`.
11. **[C]** POSTs response with status 1 (Event Received) to the Server for the SY `DERControl`.
12. **[C]** Applies the SY `DERControl` at the correct start time.
13. **[C]** POSTs response with status 2 (Event Started) for the SY event at start time.
14. **[C]** Periodically GETs the SP `DERControlList`.
15. **[C]** GETs the SP `DERControl`.

16. **[C]** POSTs response with status 1 (Event Received) to the Server for the SP `DERControl`.
17. **[C]** At the start time of the SP `DERControl`, POSTs response with status 7 (Event Superseded) to the Server for the SY `DERControl`.
18. **[C]** Applies the SP `DERControl` at the correct start time for the correct duration.
19. **[C]** POSTs response with status 2 (Event Started) for the SP event at correct start time.
20. **[C]** POSTs response with status 3 (Event Completed) for the SP event after duration has elapsed.
21. **[C]** Applies the SP `DefaultDERControl` after the completion of the event because it has a higher priority than the SY `DefaultDERControl`.

Pass/Fail Criteria

- **[C, S]** Client GETs SY `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs `SYDefaultDERControl` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DefaultDERControl` with a response code of `200 OK`.
- **[C]** Client applies the SP `DefaultDERControl` because there are no events active and it has a higher priority than the SY `DefaultDERControl`.
- **[C, S]** Client GETs SY `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs the SY `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SY event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client applies the SY `DERControl` at the correct start time.
- **[C, S]** Client POSTs response with status 2 (Event Started) for the SY event at start time. Server receives the responses and sends a response code of `201 Created`.
- **[C, S]** Client GETs the SP `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SP event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C, S]** Client, at the start time of the SP `DERControl`, POSTs response with status `7` (Event Superseded) to the Server for the SY `DERControl`. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client applies the SP `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client POSTs response with status 2 (Event Started) for the SP event at start time. Server receives the responses and sends a response code of `201 Created`.
- **[C, S]** Client POSTs response with status 3 (Event Completed) for the SP event after duration has elapsed. Server receives the responses and sends a response code of `201 Created`.

- **[C]** Client applies the SP `DefaultDERControl` after the event has completed because it has a higher priority than the SY `DefaultDERControl`.

BASIC-024 - Event - 2 DERP, 2 DDERC, 2 Overlapping Independent DERC - System DERC followed by Service Point DERC before Start of System DERC [C, A, S]

Purpose

The event (2 DERP, 2 DDERC, 2 overlapping independent DERC – system DERC followed by service point DERC before start of system DERC) test verifies the event handling of two overlapping events using independent DER controls. Because the controls are independent, the Client should execute both controls.

Note: With the change in S1 to make `DERControl` independent, `CurrentDERProgram` no longer has meaning. Errata will be added to the S1 comments and addressed in the next IEEE 2030.5 Technical Working Group. For now, do not use the `CurrentDERProgram` resource.

Setup

1. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`.
2. **[S]** Create a SY `DERControl` (for example, `opModFixedPFInjectW`) with a start time of two minutes from now and duration of two minute.
3. **[S]** Create a `DefaultDERControl` for a different control (for example, `opModFixedW`) for the SP `DERProgram`.
4. **[S]** Create a SP `DERControl` for a different control (for example, `opModFixedW`) with a start time of three minutes from now with duration of one minute.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** GETs the SY `DERProgramList`.
3. **[C]** GETs the SY `DERProgram`.
4. **[C]** GETs the SY `DefaultDERControl`.
5. **[C]** Applies the SY `defaultDERControl`.
6. **[C]** GETs the SP `DERProgramList`.
7. **[C]** GETs the SP `DERProgram`.
8. **[C]** GETs the SP `DefaultDERControl`.
9. **[C]** Applies the SP `DefaultDERControl`.
10. **[C]** Periodically GETs the SY `DERControlList`.
11. **[C]** Periodically GETs the SP `DERControlList`.
12. **[C]** GETs the SY `DERControl`.
13. **[C]** POSTs response with status 1 (Event Received) to the Server for the SY `DERControl`.
14. **[C]** Applies the SY `DERControl` at the correct start time for the correct duration.
15. **[C]** POSTs response with status 2 (Event Started) at start time for the SY `DERControl`.
16. **[C]** GETs the SP `DERControl`.

17. **[C]** POSTs response with status 1 (Event Received) to the Server for the SP `DERControl`.
18. **[C]** Applies the SP `DERControl` at the correct start time for the correct duration.
19. **[C]** POSTs response with status 2 (Event Started) at start time for the SP `DERControl`.
20. **[C]** POSTs response with status 3 (Event Completed) at the completion of the SY `DERControl`.
21. **[C]** Applies the SY `DefaultDERControl` after the completion of the event.
22. **[C]** POSTs response with status 3 (Event Completed) at the completion of the SP `DERControl`.
23. **[C]** Applies the SP `DefaultDERControl` after the completion of the event.

Pass/Fail Criteria

- **[C, S]** Client GETs SY `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs `SYDERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DefaultDERControl` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DefaultDERControl` with a response code of `200 OK`.
- **[C, S]** Client applies the SY `DefaultDERControl` because there are no events active.
- **[C, S]** Client applies the SP `DefaultDERControl` because there are no events active.
- **[C, S]** Client GETs SY `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs the SY `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SY event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client applies the SY `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client POSTs response with status 2 (Event Started) for the SY event at start time. Server receives the responses and sends a response code of `201 Created`.
- **[C, S]** Client POSTs response with status 3 (Event Completed) for the SY event after duration has elapsed. Server receives the responses and sends a response code of `201 Created`.
- **[C]** Client applies the SY `DefaultDERControl` after the event has completed.
- **[C, S]** Client GETs the SP `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SP event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client applies the SP `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client POSTs response with status 2 (Event Started) for the SP event at start time. Server receives the responses and sends a response code of `201 Created`.

- **[C, S]** Client POSTs response with status 3 (Event Completed) for the SP event after duration has elapsed. Server receives the responses and sends a response code of `201 Created`.
- **[C]** Client applies the SP `DefaultDERControl` after the event has completed.

BASIC-025 - Event - 2 DERP, 2 DDERC, 2 Overlapping Independent DERC - Service Point DERC followed by System DERC [C, A, S]

Purpose

The event (2 DERP, 2 DDERC, 2 overlapping independent DERC - service point DERC followed by system DERC) test verifies the event handling of two overlapping events using independent DER controls. Because the controls are independent, the EUT should execute both controls.

Note: With the change in S1 to make `DERControl` independent, `CurrentDERProgram` no longer has meaning. Errata will be added to the S1 comments and addressed in the next IEEE 2030.5 Technical Working Group. For now, do not use the `CurrentDERProgram` resource.

Setup

1. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`.
2. **[S]** Create a SY `DERControl` (for example, `opModFixedPFInjectW`) with a start time of three minutes from now and duration of two minute.
3. **[S]** Create a `DefaultDERControl` for a different control (for example, `opModFixedW`) for the SP `DERProgram`
4. **[S]** Create a SP `DERControl` for a different control (for example, `opModFixedW`) in the `DERProgram` with a start time of two minutes from now with duration of two minute.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** GETs the SY `DERProgramList`.
3. **[C]** GETs the SY `DERProgram`.
4. **[C]** GETs the SY `DefaultDERControl`.
5. **[C]** Applies the SY `DefaultDERControl` because there is no active event.
6. **[C]** GETs the SP `DERProgramList`.
7. **[C]** GETs the SP `DERProgram`.
8. **[C]** GETs the SP `DefaultDERControl`.
9. **[C]** Applies the SP `DefaultDERControl` because there is no active event.
10. **[C]** Periodically GETs the SY `DERControlList`.
11. **[C]** Periodically GETs the SP `DERControlList`.
12. **[C]** GETs the SY `DERControl`.
13. **[C]** POSTs response with status 1 (Event Received) to the Server for the SY `DERControl`.
14. **[C]** GETs the SP `DERControl`.
15. **[C]** POSTs response with status 1 (Event Received) to the Server for the SP `DERControl`.
16. **[C]** At the start time of the SP `DERControl`, applies the SP `DERControl`.

17. **[C]** POSTs response with status 2 (Event Started) at start time for the SP `DERControl`.
18. **[C]** At the start time of the SY `DERControl`, applies the SY `DERControl`.
19. **[C]** POSTs response with status 2 (Event Started) at start time for the SY `DERControl`.
20. **[C]** At the completion of the SP `DERControl`, applies the SP `DefaultDERControl`.
21. **[C]** POSTs response with status 3 (Event Completed) at the completion time of the SP `DERControl`.
22. **[C]** At the completion of the SY, applies the SY `DefaultDERControl`.
23. **[C]** POSTs response with status 3 (Event Completed) at the completion time of the SY `DERControl`.

Pass/Fail Criteria

- **[C, S]** Client GETs SY `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DefaultDERControl` with a response code of `200 OK`.
- **[C]** Client applies SY `DefaultDERControl` because there are no events active.
- **[C, S]** Client GETs SP `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DefaultDERControl` with a response code of `200 OK`.
- **[C]** Client applies the SP `DefaultDERControl` because there are no events active.
- **[C, S]** Client GETs SY `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs the SY `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SY event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C, S]** Client GETs the SP `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SP event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client, at the start time of the SP `DERControl`, applies SP `DERControl`.
- **[C, S]** Client, at the start time of the SP `DERControl`, POSTs response with status 2 (Event Started) to the Server for the SP `DERControl`. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client, at the start time of the SY `DERControl`, applies SY `DERControl`.
- **[C, S]** Client, at the start time of the SY `DERControl`, POSTs response with status 2 (Event Started) to the Server for the SY `DERControl`. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client, at the completion of the SP `DERControl`, applies SP `DefaultDERControl`.
- **[C, S]** Client, at the completion of the SP `DERControl`, POSTs response with status 3 (Event Completed) to the Server for the SP `DERControl`. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client, at the completion of the SY `DERControl`, applies SY `DefaultDERControl`.

- **[C, S]** Client, at the completion of the SY `DERControl`, POSTs response with status 2 (Event Completed) to the Server for the SY `DERControl`. Server receives the response and sends a response code of `201 Created`.

BASIC-026 - Event - 2 DERP, 2 DDERC, 2 Overlapping Independent DERC - System DERC followed by Service Point DERC after Start of System Event [C, A, S]

Purpose

The event (2 DERP, 2 DDERC, 2 overlapping independent DERC - system DERC followed by service point DERC after start of system event) test verifies the event handling of two overlapping events using independent DER controls. Because the controls are independent, the Client should execute both controls.

Note: With the change in S1 to make `DERControl` independent, `CurrentDERProgram` no longer has meaning. Errata will be added to the S1 comments and addressed in the next IEEE 2030.5 Technical Working Group. For now, do not use the `CurrentDERProgram` resource.

Setup

1. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`
2. **[S]** Create a SY `DERControl` (for example, `opModFixedPFInjectW`) with a start time of one minute from now and duration of four minutes.
3. **[S]** Create a `DefaultDERControl` for a different control (for example, `opModFixedW`) for the SP `DERProgram`.
4. **[S]** After the SY has started, create a SP `DERControl` for the different control, `opModFixedW`, with a start time of two minutes from now and a two-minute duration.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** GETs the SY `DERProgramList`.
3. **[C]** GETs the SY `DERProgram`.
4. **[C]** GETs the SY `DefaultDERControl`.
5. **[C]** Applies the SY `DefaultDERControl` because there is no active event.
6. **[C]** GETs the SP `DERProgramList`.
7. **[C]** GETs the SP `DERProgram`.
8. **[C]** GETs the SP `DefaultDERControl`.
9. **[C]** Applies the SP `DefaultDERControl` because there is no active event.
10. **[C]** Periodically GETs the SY `DERControlList`.
11. **[C]** Periodically GETs the SP `DERControlList`.
12. **[C]** GETs the SY `DERControl`.
13. **[C]** POSTs response with status 1 (Event Received) to the Server for the SY `DERControl`.
14. **[C]** Applies the SY `DERControl` at the correct start time for the correct duration.
15. **[C]** POSTs response with status 2 (Event Started) for this SY event at start time.
16. **[C]** GETs the SP `DERControl`.

17. **[C]** POSTs response with status 1 (Event Received) to the Server for the SP `DERControl`.
18. **[C]** Applies the SP `DERControl` at the correct start time for the correct duration.
19. **[C]** POSTs response with status 2 (Event Started) for the SP event at start time.
20. **[C]** POSTs response with status 3 (Event Completed) for the SP event after duration has elapsed.
21. **[C]** Applies the SP `DefaultDERControl` after the completion of the event.
22. **[C]** POSTs response with status 3 (Event Completed) for the SY event after duration has elapsed.
23. **[C]** Applies the SY `DefaultDERControl` after the completion of the event.

Pass/Fail Criteria

- **[C, S]** Client GETs SY `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SY `DefaultDERControl` with a response code of `200 OK`.
- **[C]** Client applies the SY `DefaultDERControl` because there are no events active.
- **[C, S]** Client GETs SP `DERProgramList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERProgram` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DefaultDERControl` with a response code of `200 OK`.
- **[C]** Client applies the SP `DefaultDERControl` because there are no events active.
- **[C, S]** Client GETs SY `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs SP `DERControlList` with a response code of `200 OK`.
- **[C, S]** Client GETs the SY `DERControl` with a response code of `200 OK`.
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SY event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client applies the SY `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client POSTs response with status 2 (Event Started) for the SY event at start time. Server receives the responses and sends a response code of `201 Created`.
- **[C, S]** Client POSTs response with status 3 (Event Completed) for the SY event after duration has elapsed. Server receives the responses and sends a response code of `201 Created`.
- **[C]** Client applies the SY `DefaultDERControl` after the event has completed
- **[C, S]** Client GETs the SP `DERControl` with a response code of `200 OK`
- **[C, S]** Client POSTs response with status 1 (Event Received) for the SP event to the Server. Server receives the response and sends a response code of `201 Created`.
- **[C]** Client applies the SP `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client POSTs response with status 2 (Event Started) for the SP event at start time. Server receives the responses and sends a response code of `201 Created`.

- **[C, S]** Client POSTs response with status 3 (Event Completed) for the SP event after duration has elapsed. Server receives the responses and sends a response code of `201 Created`.
- **[C]** Client applies the SP `DefaultDERControl` after the event has completed

BASIC-027 – Alarms [C, A, S]

Purpose

The alarms test verifies tht the Clients sends alarm-related information to the IEEE 2030.5 server in the time frame required by the CSIP and IEEE 2030.5 standards.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Pre-register an `EndDevice` instance for the Client device, including all following attributes. For example, `SFDI/LFDI`, `FunctionSetAssignmentsListLink`, `RegistrationLink/PIN`, `LogEventListLink`.
3. **[C]** Confirm that an error can be raised and detected which shall trigger a `LogEvent` to be sent to the EUT-Server.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Using the `EndDevice` instance, find the `LogEventListLink` and its attributes, to use the link for the rest of this test.
3. **[C]** Prepare a `LE_GEN_SOFTWARE` general `LogEvent` (listed in Table 34 - General Log Events) with associated attributes in the `LogEvent` payload to be posted, such as `createdDateTime`, `functionSet` (shall be zero), `logEventCode`, `logEventID`, `logEventPEN` and `profileID`. Do an HTTP POST using this payload to the `LogEventListLink` from the previous step.
4. **[S]** Successfully receive the HTTP POST from the Client, process the information, save the information and create a link instance to use in the HTTP POST response as the location header. If successful, respond to the HTTP POST by issuing an HTTP `201` response and the Location header. If unsuccessful, issue the appropriate HTTP response code and any other response body, if relevant.
5. **[TC]** Repeat steps 2 and 3 five times to test the Server ability to handle at least five `LogEvent` instances.
6. **[TC]** Perform an HTTP GET operation on the `LogEventListLink` with query string parameter `?l=255`.
7. **[S]** Successfully respond to the HTTP GET on the `LogEventListLink` by sending the list contents that meet the provided query string parameters.
8. **[TC]** Verify the response body of the HTTP GET on the `LogEventListLink` carries at least `LogEvent` instances and they are time ordered (most recent listed third). There can be more than five because the Server may have other instances already included.

Pass/Fail Criteria

- **[C, S]** Client was able to successfully find the `LogEventListLink` in its `EndDevice` instance and its attributes. The Server included a conformant and correct set of attributes for the `LogEventListLink` instance for this client.
- **[C, S]** Client was able to prepare a conformant `LogEvent` payload to use in an HTTP POST to the Server using the test step described above and issued a successful HTTP POST using such payload.
- **[C, S]** Server was able to successfully receive the HTTP POST of the `LogEvent` payload from the Client and processed it successfully to issue an HTTP `201` response and Location header.
- **[TC, S]** Client and Server were able to successfully repeat the HTTP POST of five unique `LogEvent` payloads and meet all criteria listed for such steps in this test.
- **[TC, S]** Client was able to successfully perform an HTTP GET using the query string parameter on the client `LogEventListLink`.
- **[TC, S]** Server was able to successfully respond to the HTTP GET on the `LogEventListLink` by sending a correct HTTP response code and response body to the Client.
- **[TC, S]** Client was able to verify the response body of the HTTP GET on the `LogEventListLink` carries at least `LogEvent` instances and they are time ordered (most recent listed third).

BASIC-028 - Inverter Status [C, A, S]

Purpose

The inverter status test verifies that the Client sends DER information to the IEEE 2030.5 server, including `DERStatus` and `DERAvailability`.

Setup

1. **[S]** Verify a `DeviceCapability` resource exists on the Server, which includes a link to `EndDeviceListLink` and its subordinate resources.
2. **[S]** Pre-register an `EndDevice` instance for the Client device, including all following attributes. For example, `SFDI/LFDI`, `FunctionSetAssignmentsListLink`, `RegistrationLink`/`PIN`, `LogEventListLink`, `DERListLink`.
3. **[C]** Confirm it is capable of reporting status against one (or more) of the `DERStatus` elements.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Using the `EndDevice` instance retrieved from the previous test step, find the `DERListLink` and its attributes to use the link for the rest of this test.
3. **[C]** Do a series of HTTP GETs on the `DERListLink` to retrieve `DERList` subordinate resources and validate the attributes associated with `DERCapability`, `DERSettings`, `DERStatus`, and other attributes reflect its last updated state.
4. **[C]** Prepare a `DERStatus` payload by updating its status with the following elements with associated current values.
    - For Inverter (Power generating) device types:
        - `genConnectStatus`
        - `inverterStatus`
        - `localControlModeStatus`
        - `manufacturerStatus`
        - `operationalModeStatus`
        - `readingTime`
    - For Storage device types:
        - `localControlModeStatus`
        - `manufacturerStatus`
        - `operationalModeStatus`
        - `readingTime`
        - `stateOfChargeStatus`
        - `storageModeStatus`
        - `storConnectStatus`

5. **[C]** Using the prepared `DERStatus` payload that reflects its current state, perform an HTTP PUT operation on the `DERStatusLink` href.
6. **[S]** Successfully respond to the HTTP PUT operation from the previous step and update its internal state and `DERStatusLink` for this Client `EndDevice` instance.

Pass/Fail Criteria

- **[C, S]** Client was able to find the `DERListLink` in its `EndDevice` instance retrieved from the Server. Client received the correct HTTP response code and conformant payload.
- **[C, S]** The Client successfully did a series of HTTP GETs on the `DERListLink` to retrieve `DERList` subordinate resources and validated the attributes associated with `DERCapability`, `DERSettings`, `DERStatus` and other attributes reflect its last updated state. Server successfully responded to the HTTP GET requests on the `DERListLink` and its subordinate resources.
- **[C]** The Client successfully prepared a `DERStatus` payload by updating its status with the following elements with associated current values.
  - For inverter (Power generating) device types:
    - `genConnectStatus`
    - `inverterStatus`
    - `localControlModeStatus`
    - `manufacturerStatus`
    - `operationalModeStatus`
    - `readingTime`

    - For storage device types:
        - `localControlModeStatus`
        - `manufacturerStatus`
        - `operationalModeStatus`
        - `readingTime`
        - `stateOfChargeStatus`
        - `storageModeStatus`
        - `storConnectStatus`
- **[C, S]** The Client successfully did an HTTP PUT on the `DERStatus` payload using the prepared payload that reflected its current state from the previous step.
- **[C, S]** Server successfully responded to the HTTP PUT on the `DERStatus` payload by sending it correct 204 HTTP response code and reflected the information included in the `DERStatus` to the `EndDevice` associated instance.

BASIC-029 - Inverter Meter Reading [C, A, S]

Purpose

The inverter meter reading test verifies that the Client sends metered reading values using IEEE 2030.5 mirrored metering POST requests.

Setup

1. Server Device Capabilities resource shall include `MirrorUsagePointListLink` for Client devices to post mirrored metering readings and the Server shall be prepared to process incoming mirrored metering readings, including creation of mirrored `UsagePoint` instances.
2. Client shall be prepared to post various CSIP required metered readings as described in Section 5.7.1 Meter Data in the CSIP guide. For example, Real Power, Reactive power, Frequency, and Voltage. It shall also know its own `LFDI` value to include it in the Mirrored Meter readings.

Procedure

1. **[T]** Record the Client/Server communications.
2. Locate the `MirrorUsagePointListLink` URI provided by the Server device capabilities response payload by running the **Error! Reference source not found.** test.
3. **[C]** The Client shall prepare the `ReadingType` for each type of reading it intends to send to the Server and assign it a specific `mRID`, which it associates subsequent `MirrorMeterReading` instances. The meter `ReadingType` that must be defined in the post `MirrorUsagePoint` are:
   - Real Power (W)
   - Reactive Power (VAr)
   - Frequency (Hz)
   - Voltage

   Real Power (W) defined as `MirrorMeterReading`/`ReadingType` example:

```
<MirrorMeterReading>
  <mRID>xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx</mRID>
  <description>Real Power(W)</description>
  <ReadingType>

  <accumulationBehaviour>12</accumulationBehaviour>
          <commodity>1</commodity>
          <dataQualifier>0</dataQualifier>
          <flowDirection>1</flowDirection>
          <kind>37</kind>
          <phase>0</phase>
          <powerOfTenMultiplier>0</powerOfTenMultiplier>
          <uom>38</uom>
  </ReadingType>
```

```
</MirrorMeterReading>
```

Refer to the CSIP Implementation Guide and IEEE 2030.5 standard for more details. The metered data type listed above shall be defined using separate `MirrorMeterReading`/`ReadingType` payloads.

4. **[C]** Using the `MirrorUsagePointListLink` URI from step 1, perform an HTTP POST operation on this URI using the `MirrorUsagePoint` and Real Power (W) `MirrorMeterReading`/`ReadingType` now, the Client is setting up a `MirrorUsagePoint` (with a unique `mRID`) with the `MirrorMeterReading`/`ReadingType` defined in the previous step and not posting actual metered data itself.
5. **[S]** Process the requested Mirrored`UsagePoint` POST from the Client, including creation of the mirror `UsagePoint`. If successful, the Server shall respond to the Client requested with a `201 Created` with a Location header that specifies the URI of the matching `MirrorUsagePoint`.
6. Repeat steps 3 and 4 by performing an HTTP POST on remaining metered data types. For example, Reactive Power (VAr), Frequency (Hz), Voltage. This series of HTTP POST shall result in HTTP `201 Created` for the unique metered data type and a Location header to the mirrored `UsagePoint` meter reading instances. For example, `/upt/1/mr/1`, `/upt/1/mr/2`, and others.
7. **[C]** Prepare the metered data to be used for the mirrored meter data set, including Real Power. For each metered data type, the `mRID` of the `MirrorMeterReading` shall match the `mRID` used during `ReadingType` setup in previous steps. After the Real Power (W) metered data is prepared, perform an HTTP POST operation using the `MirrorUsagePointListLink` URI and `MirrorMeterReading`/Reading body.
8. **[S]** Process the requested Mirrored`UsagePoint` POST from the Client, including creation of the mirror `UsagePoint`. If successful, the Server shall respond to the Client requested with a `204 No Content`.
9. **[C]** Repeat steps 6 and 7 for the remaining metered data types: Apparent Power, Reactive Power, Power Factor, Voltage, Current and other readings as required by the utility interconnection handbook.
10. **[C]** Using the `UsagePoint` meter reading instance locations from step 5, perform an HTTP GET operation on the locations and verify the data returned matches the sent data from the previous HTTP POST operation.

Pass/Fail Criteria

- **[C]** The Client successfully prepared the various metered usage data as described in the CSIP Implementation Guide by creating the correct `MirrorMeterReading` and `ReadingType` definitions.

- **[C, S]** The Client successfully did an HTTP POST using the `MirrorUsagePoint` and `MirrorMeterReading`/`ReadingType` definitions for Real Power (W).
- **[C, S]** Server successfully responded to the previous HTTP POST on the `MirrorUsagePoint` and returned an HTTP `201 Created` and a Location header for the MUP instance. The Client successfully received this set of responses from the Server because of its HTTP POST.
- **[C, S]** Client and Server successfully repeated the HTTP POST on remaining metered data types. For example, Reactive Power (VAr), Frequency (Hz), Voltage. This series of HTTP POST resulted in HTTP `201 Created` for the unique metered data type and a Location header to the mirrored `UsagePoint` meter reading instances. For example, /upt/1/mr/1, /upt/1/mr/2, and others.
- **[C, S]** The Client successfully prepared the metered data to be used for the mirrored meter data set, including Real Power. If the Client is an aggregator, it summed up these values from individual DER devices it manages (see CSIP guide for further information). For each metered data type, the `mRID` of the `MirrorMeterReading` matched the `mRID` used during `ReadingType` setup in previous steps. After the Real Power (W) metered data was prepared, The Client successfully did an HTTP POST using the `MirrorUsagePointListLink` URI and `MirrorMeterReading`/Reading body.
- **[C, S]** Server successfully responded to the requested Mirrored`UsagePoint` POST from the Client which included creation of the mirror `UsagePoint` by responding with HTTP `204 No Content` response.
- **[C, S]** Client and Server successfully repeated steps 6 and 7 for the remaining metered data types: Reactive Power, Power Factor, Frequency, Voltage as required by the utility interconnection handbook.
- **[C, S]** The Client successfully used the `UsagePoint` meter reading instance locations from step 5, did an HTTP GET on the locations and verified the data returned matches the sent data from the previous HTTP POST operation.

8 Utility Server Aggregator Model Tests

This test section covers the 5.5 Utility Server Operation - Aggregator Model section of the CSIP guide. These tests target the utility server ability to create and manage the IEEE 2030.5 resources required to support DER aggregators communicating with the utility.

The utility server EUT shall have completed the server-related tests included in the Communication Fundamentals Tests, Core Function Set Tests, and Basic Functions Tests sections of this document before executing the tests in this section.

These tests assume the aggregator manages four `EndDevice` entities (EDA1, EDA2, EDB1, and EDB2) shown in the following topology:

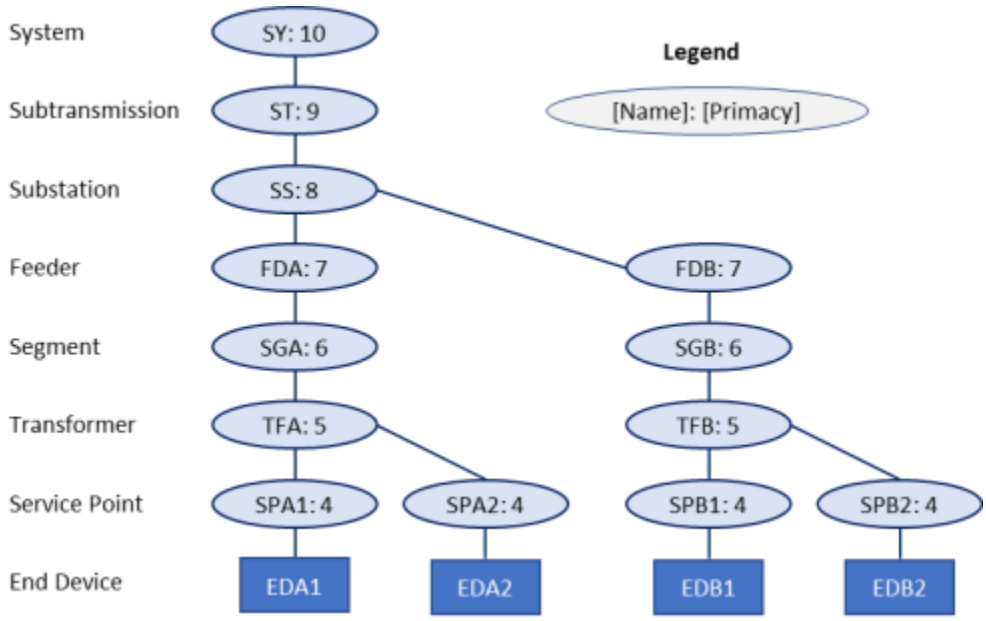

Figure 15 Aggregator End Device Topology for Utility/Aggregator Tests

The IEEE 2030.5 standard permits many ways to create the topology, using `FunctionSetAssignments` and the associated `DERProgram`, but requires a `DERProgram` associated with each group node used to send DER controls to all devices connected to that node.

UTIL-001 - Utility Server Startup Configuration Group Assignment of Inverters [S]

Purpose

The utility server startup configuration group assignment of inverters test verifies that the utility IEEE 2030.5 server and its aggregator client devices can be assigned to one or more groups based on the distribution topology. This test shall include test scenarios using various grouping levels, including a maximum level of nine with uniquely assigned `DERPrograms` for each level.

Setup

1. **[S]** Create the topology show in Figure 15 Aggregator End Device Topology for Utility/Aggregator Tests.
2. **[S]** Configure the server to create a DER Program associated with each node with the corresponding primacy value.
3. **[S]** Assign inverter EDA1 to node SPA1. Assign EDA2 to node SPA2. Assign EDB1 to node SPB1. Assign EDB2 to node SPB2.
4. **[S]** Create an `EndDeviceList` for the aggregator Client
5. **[S]** Create an `EndDevice` instance for the aggregator with:
    - `SFDI`/`LFDI` - aggregator `SFDI` and `LFDI`
    - `SubscriptionListLink`
    - `LogEventListLink`
6. **[S]** Create an `EndDevice` instance for EDA1, EDA2, EDB1 and EDB2 with:
    - `SFDI`/`LFDI` - `EndDevice SFDI` and `LFDI`
    - `FunctionSetAssignmentsListLink`
    - `DERListLink`
    - `LogEventListLink`

Procedure

1. **[T]** Record the Client/Server communications.
2. **[TC]** GET the `DeviceCapability` resource. For example, `/dcap`.
3. **[TC]** GET the `EndDeviceList` resource (for example, `/edev`) using the `DeviceCapability` link.
4. **[TC]** GET all `EndDevice` entities from the `EndDeviceList` resource (for example, /edev) using the `DeviceCapability` links.
5. **[TC]** For each non-aggregator `EndDevice` instance, GET all the `FunctionSetAssignments` from its `FunctionSetAssignmentsList`.
6. **[TC]** For each `FunctionSetAssignment`, GET all the `DERPrograms` from its `DERProgramList`.

Pass/Fail Criteria

- **[TC]** GETs the `DeviceCapability` resource. For example, `/dcap`.

- **[TC]** GETs the `EndDeviceList` resource using the link from `DeviceCapability`. For example, `/edev`.
- **[TC]** GETs all `EndDevice` entities from the `EndDeviceList` resource using the link from `DeviceCapability`s. For example, `/edev`. There should be an `EndDevice` instance for the aggregator, EDA1, EDA2, EDB1, and EDB2.
- **[TC]** For each non-aggregator `EndDevice` instance, GETs the `FunctionSetAssignments` from its `FunctionSetAssignmentsList`.
- **[TC]** For each `FunctionSetAssignment`, GETs the `DERPrograms` from its `DERProgramList`.
- **[TC]** Each `EndDevice` should be assigned a link to all the DER programs associated with its parent nodes. Verify this is the case.

UTIL-002 - Utility - Aggregator Operations Commissioning [A, S]

Purpose

The utility-aggregator operations commissioning test verifies that the aggregator can interact with its utility server to be commissioned using the topology-based grouping for its managed inverters. This test shall include test scenarios that test the commissioning aspects of the operation, including discovery, secure connection, and retrieval of device information for all managed downstream inverters.

Setup

Perform the test setup from the UTIL-001 - Utility Server Startup Configuration Group Assignment of Inverters test to configure the Utility Server.

Procedure

1. **[C]** Retrieve the `DeviceCapability` resource from the Server using the supported HTTP and IP address and find the `EndDeviceListLink` element.
2. **[C]** Perform an HTTP GETs on the `EndDeviceListLink` URI and GETs all the `EndDevice` entities in the list.
3. **[C]** Verifies there is an `EndDevice` instance with an `SFDI`/`LFDI` that matches its own (aggregator) `SFDI`/`LFDI`.
4. **[C]** Perform an HTTP GET operation on the aggregator `RegistrationLink` href to find the PIN value for the Client device.
5. **[C]** Process the returned Registration resource and search for the PIN element and find its value. Verify the PIN value is the same PIN value the Client device has preregistered.
6. **[C]** For all other `EndDevice` instances (EDA1, EDA2, EDB1, and EDB2), find the `DERListLink` information. The `DERListLink` resources have information about the DER device state, capabilities and settings. Do an HTTP PUT on the `DERListLink` using the href attribute and updated values for `DERCapabilities`, `DERSettings`, `DERStatus` or `DERAvailability`.

Pass/Fail Criteria

- **[C, S]** Client retrieves the `DeviceCapability` resource from the Server using the supported HTTP and IP address and finds the `EndDeviceListLink` element.
- **[C, S]** Client performs an HTTP GETs on the `EndDeviceListLink` URI and GETs all the `EndDevice` s in the list.
- **[C]** Client verifies there is an `EndDevice` instance with an `SFDI`/`LFDI` that matches its own (aggregator) `SFDI`/`LFDI`.
- **[C, S]** Client performs an HTTP GET on the aggregator `RegistrationLink` href to find the PIN value for the Client device.
- **[C]** Client processes the returned Registration resource and search for the PIN element and finds its value. Verifies the PIN value is the same PIN value the Client device has preregistered.

- **[C, S]** Client, for all other `EndDevice` instances (EDA1, EDA2, EDB1, and EDB2), finds the `DERListLink` information. The `DERListLink` resources have information about the DER device state, capabilities and settings. Performs an HTTP PUT on the `DERListLink` using the href attribute and updated values for `DERCapabilities`, `DERSettings`, `DERStatus` or `DERAvailability`.

UTIL-003 - Utility - Aggregator Operations Group Assignments Retrieval [A, S]

Purpose

The utility-aggregator operations group assignments retrieval test verifies that the aggregator can interact with its utility server to retrieve the topology-based grouping for its managed inverters. This test shall include test scenarios that test aggregator conformance to get and process all relevant `FunctionSetAssignments` and `EndDevice` information for all managed inverters, including the priority determination, if there are FSA-assigned conflicting `DERPrograms`.

Setup

Perform the test setup from the UTIL-001 - Utility Server Startup Configuration Group Assignment of Inverters test to configure the Utility Server.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** For each non-aggregator `EndDevice` instance (EDA1, EDA2, EDB1, and EDB2), find the `FunctionSetAssignmentsListLink` information and GET all the `FunctionSetAssignments`.
3. **[C]** For each non-aggregator `EndDevice` instance (EDA1, EDA2, EDB1, and EDB2), GET all `DERPrograms` from the `DERProgramListLink` in all the `FunctionSetAssignments`.
4. **[C]** For each non-aggregator `EndDevice` instance (EDA1, EDA2, EDB1, and EDB2), aggregator subscribes to all `DERPrograms` assigned to the `DERPrgramList,` so it is notified on changes in primacy.
5. **[C]** For each `DERProgram`, the aggregator subscribes to the `DERControlList,` so it is notified of changes (for example, new `DERControls`) to the list.

Pass/Fail Criteria

- **[C, S]** Client, for each non-aggregator `EndDevice` instance (EDA1, EDA2, EDB1, and EDB2), finds the `FunctionSetAssignmentsListLink` information and GETs all the `FunctionSetAssignments`.
- **[C, S]** Client, for each non-aggregator `EndDevice` instance (EDA1, EDA2, EDB1, and EDB2), GETs all `DERPrograms` from the `DERProgramListLink` in all the `FunctionSetAssignments`.
- **[C, S]** Client, for each non-aggregator `EndDevice` instance (EDA1, EDA2, EDB1, and EDB2), aggregator subscribes to all `DERPrograms` assigned to the `DERProgramList` so it is notified on changes in primacy.
- **[C, S]** Client, for each `DERProgram`, the aggregator subscribes to the `DERControlList,` so it is notified of changes to the list. For example, new `DERControls`.

UTIL-004 - Utility - Aggregator Operations DER Retrieval [A, S]

Purpose

The utility-aggregator operations DER retrieval test verifies that the aggregator can interact with its utility server to get the DERProgram/DERControl and associated information, such as DERCurve, DefaultDERControl, and ActiveDERControl. This test shall include test scenarios that test aggregator conformance to get and process relevant DERProgram/DERControl information for all managed inverters, including scheduling of appropriate DER events as applicable.

Setup

1. **[S]** Create the topology show in Figure 15 Aggregator End Device Topology for Utility/Aggregator Tests.
2. **[S]** Configure the server to create a DER Program associated with each node with the corresponding primacy value.
3. **[S]** Assign inverter EDA1 to node SPA1. Assign EDA2 to node SPA2. Assign EDB1 to node SPB1. Assign EDB2 to node SPB2.
4. **[S]** Create an EndDeviceList for the aggregator Client
5. **[S]** Create an EndDevice instance for the aggregator with:
    - SFDI/LFDI - aggregator SFDI and LFDI
    - SubscriptionListLink
    - LogEventListLink
6. **[S]** Create an EndDevice instance for EDA1, EDA2, EDB1 and EDB2 with:
    - SFDI/LFDI - EndDevice SFDI and LFDI
    - FunctionSetAssignmentsListLink
    - DERListLink
    - LogEventListLink
7. **[C]** For each non-aggregator EndDevice instance (EDA1, EDA2, EDB1, and EDB2), find the FunctionSetAssignmentsListLink information and GET all the FunctionSetAssignments.
8. **[C]** For each non-aggregator EndDevice instance (EDA1, EDA2, EDB1, and EDB2), GET all DERPrograms from the DERProgramListLink in all the FunctionSetAssignments.
9. **[C]** For each non-aggregator EndDevice instance (EDA1, EDA2, EDB1, and EDB2), aggregator subscribes to all DERPrograms assigned to the DERProgramList, so it is notified on changes in primacy.
10. **[C]** For each DERProgram, the aggregator subscribes to the DERControlList, so it is notified of changes (for example, new DERControls) to the list.

Procedure

1. **[T]** Record the Client/Server communications.

2. **[S]** Create a `DERControl` (for example, `opModFixedPFInjectW`) on node SPA1 with a start time of two minutes from now and duration of two minute.
3. **[S]** Notifies Client of change to SPA1 `DERControlList`.
4. **[C]** Receive notification and GETs the `DERControlList` and GETs the newly created `DERControl`.
5. **[C]** POSTs a `received` response to the Server for EDA1.
6. **[C]** At the start time of the event, POSTs a `started` response to the Server for EDA1.
7. **[C]** At the end time of the event, POSTs a `completed` response to the Server for EDA1.
8. Repeat steps 1 to 6 for nodes SPA2, SPB1, and SPB2 and the corresponding `EndDevices` EDA2, EDB1, and EDB2.
9. **[S]** Create a `DERControl` (for example, `opModFixedPFInjectW`) on node FDA with a start time of two minutes from now and duration of two minute.
10. **[S]** Notifies Client of change to FDA `DERControlList`.
11. **[C]** Receive notification and GETs the `DERControlList` and GETs the newly created `DERControl`.
12. **[C]** POSTs a `received` response to the Server for EDA1 and EDA2.
13. **[C]** At the start time of the event, POSTs a `started` response to the Server for EDA1 and EDA2.
14. **[C]** At the end time of the event, POSTs a `completed` response to the Server for EDA1 and EDA2.
15. Repeat steps 8 to 13 for node FDB and the corresponding `EndDevice` s EDB1 and EDB2.
16. **[S]** Create a `DERControl` (for example, `opModFixedPFInjectW`) on node SY with a start time of two minutes from now and duration of two minute.
17. **[S]** Notifies Client of change to SY `DERControlList`.
18. **[C]** Receive notification and GETs the `DERControlList` and GETs the newly created `DERControl`.
19. **[C]** POSTs a `received` response to the Server for EDA1, EDA2, EDB1, and EDB2.
20. **[C]** At the start time of the event, POSTs a `started` response to the Server for EDA1, EDA2, EDB1, and EDB2.
21. **[C]** At the end time of the event, POSTs a `completed` response to the Server for EDA1, EDA2, EDB1, and EDB2.

Pass/Fail Criteria

- **[C, S]** Server creates a `DERControl` (for example, `opModFixedPFInjectW`) on node SPA1 with a start time of two minutes from now and duration of two minute.
- **[C, S]** Server notifies Client of change to SPA1 `DERControlList`.
- **[C, S]** Client receives notification and GETs the `DERControlList` and GETs the newly created `DERControl`.
- **[C, S]** Client POSTs a `received` response to the Server for EDA1.

- **[C, S]** Client, at the start time of the event, POSTs a `started` response to the Server for EDA1.
- **[C, S]** Client, at the end time of the event, POSTs a `completed` response to the Server for EDA1.
- **[C, S]** Repeat steps 1 to 6 for nodes SPA2, SPB1, and SPB2 and the corresponding `EndDevice` s EDA2, EDB1, and EDB2.
- **[C, S]** Server creates a `DERControl` (for example, `opModFixedPFInjectW`) on node FDA with a start time of two minutes from now and duration of two minute.
- **[C, S]** Server notifies Client of change to FDA `DERControlList`.
- **[C, S]** Client receives notification and GETs the `DERControlList` and GETs the newly created `DERControl`.
- **[C, S]** Client POSTs a `received` response to the Server for EDA1 and EDA2.
- **[C, S]** Client, at the start time of the event, POSTs a `started` response to the Server for EDA1 and EDA2.
- **[C, S]** Client, at the end time of the event, POSTs a `completed` response to the Server for EDA1 and EDA2.
- **[C, S]** Repeat steps 8 to 13 for node FDB and the corresponding `EndDevice` s EDB1 and EDB2.
- **[C, S]** Server creates a `DERControl` (for example, `opModFixedPFInjectW`) on node SY with a start time of two minutes from now and duration of two minute.
- **[C, S]** Server notifies Client of change to SY `DERControlList`.
- **[C, S]** Client receives notification and GETs the `DERControlList` and GETs the `DERControl`.
- **[C, S]** Client POSTs a `received` response to the Server for EDA1, EDA2, EDB1, and EDB2.
- **[C, S]** Client, at the start time of the event, POSTs a `started` response to the Server for EDA1, EDA2, EDB1, and EDB2.
- **[C, S]** Client, at the end time of the event, POSTs a `completed` response to the Server for EDA1, EDA2, EDB1, and EDB2.

9 Aggregator Operation Tests

This section covers the 5.6 Aggregator Operation section of the CSIP guide. These tests verify that the aggregator can interact with a utility server inconformance with the requirements of a CSIP aggregator as specified in the CSIP guide.

These tests involve:

- subscription/notification to optimize utility-aggregator interactions
- a DER event operation series, including conflict resolution

The aggregator EUT shall have completed the tests described in Communication Fundamentals Tests, Core Function Set Tests, and Basic Functions Tests before running the tests contained in this section.

AGG-001 - Aggregator Operation Subscription [A, S]

Purpose

The aggregator operation subscription test verifies the subscription/notification mechanism for the server to send updates to its aggregator clients. This test shall include scenarios to communicate changes to `EndDeviceList`, `EndDevice`, `FunctionSetAssignmentsList`, `DERControlList`, `DERProgramList` and `DERProgram` resources.

Setup

1. **[S]** Create the topology show in Figure 15 Aggregator End Device Topology for Utility/Aggregator Tests.
2. **[S]** Configure the server to create a DER Program associated with each node with the corresponding primacy value.
3. **[S]** Assign inverter EDA1 to node SPA1. Assign EDA2 to node SPA2. Assign EDB1 to node SPB1. Assign EDB2 to node SPB2.
4. **[S]** Create an `EndDeviceList` for the aggregator Client
5. **[S]** Create an `EndDevice` instance for the aggregator with:
    - `SFDI`/`LFDI` - aggregator `SFDI` and `LFDI`
    - `SubscriptionListLink`
    - `LogEventListLink`
6. **[S]** Create an `EndDevice` instance for EDA1, EDA2, EDB1 and EDB2 with:
    - `SFDI`/`LFDI` - `EndDevice SFDI` and `LFDI`
    - `FunctionSetAssignmentsListLink`
    - `DERListLink`
    - `LogEventListLink`

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Subscribe to the `EndDeviceList`.
3. **[S]** Create a new `EndDevice` instance for EDA1X and assign it to node SPA1.
4. **[S]** Send a notification to the Client because there is a change (addition) to the `EndDeviceList`.
5. **[C]** Receives the notification of the `EndDeviceList` and GETs the `EndDeviceList` and GETs the newly created EDA1X `EndDevice` instance.

Pass/Fail Criteria

- **[C, S]** Client subscribes to the `EndDeviceList`.
- **[C, S]** Server creates a new `EndDevice` instance for EDA1X and assigns it to node SPA1.
- **[C, S]** Server sends a notification to the Client because there is a change (addition) to the `EndDeviceList`.

- **[C, S]** Client receives the notification of the `EndDeviceList`, GETs the `EndDeviceList`, and GETs the newly created EDA1X `EndDevice` instance.

AGG-002 - Aggregator Event - 2 DERP, 2 DDERC, 0 DERC [A, S]

Purpose

The aggregator event (2 DERP, 2 DDERC, 0 DERC) test verifies that the aggregator executes the correct default DER control for the four end devices under its control.

Setup

1. **[S]** Perform the test setup from AGG-001.
2. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`
3. **[S]** Create a `DefaultDERControl` for the same specific control (for example, `opModFixedPFInjectW`) for the TFA `DERProgram`.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** For EDA1 and EDA2, applies the TFA `DefaultDERControl`.
3. **[C]** For EDB2 and EDB2, applies the SY `DefaultDERControl`.

Pass/Fail Criteria

- **[C]** Client, for EDA1 and EDA2, applies the TF `DefaultDERControl`. Note that there is no acknowledgment for activating a `DefaultDERControl` in the IEEE 2030.5 protocol. Verification of the activation of the `DefaultDERControl` must be done out-of-band.
- **[C]** Client, for EDB2 and EDB2, applies the SY `DefaultDERControl`. Note that there is no acknowledgment for activating a `DefaultDERControl` in the IEEE 2030.5 protocol. Verification of the activation of the `DefaultDERControl` must be done out-of-band.

AGG-003 - Aggregator Event - 1 DERP, 0 DDERC, 1 DERC [A, S]

Purpose

The aggregator event (1 DERP, 0 DDERC, 1 DERC) test verifies that the aggregator executes a single DER event on the correct end devices.

Setup

1. **[S]** Perform the test setup from AGG-001.
2. **[S]** Create a `DERControl` for a specific control (for example, `opModFixedPFInjectW`) for the TFA `DERProgram` with a start time of two minutes from now and duration of one minute.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) to the Server.
3. **[C]** Aggregator applies the TFA `DERControl` at the correct start time and duration.
4. **[C]** For EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time.
5. **[C]** For EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration has elapsed.

Pass/Fail Criteria

- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) to the Server.
- **[C]** Client aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 2 (event started) at start time.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response `3` (event completed) after elapsed duration.
- Client fails if EDB1 and/or EDB2 POSTs any responses to the TFA event.

AGG-004 - Aggregator Event - 1 DERP, 1 DDERC, 1 DERC [A, S]

Purpose

The aggregator event (1 DERP, 1 DDERC, 1 DERC) test verifies that the aggregator executes a single DER event on the correct end devices and applies the default DERC when the DER event is not active.

Setup

1. **[S]** Perform the test setup from AGG-001.
2. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the TFA `DERProgram`.
3. **[S]** Create a `DERControl` for a specific control (for example, `opModFixedPFInjectW`) for the TFA `DERProgram` with a start time of two minutes from now and duration of one minute.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
3. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) to the Server.
4. **[C]** aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
5. **[C]** For EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time.
6. **[C]** For EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration has elapsed.
7. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`.

Pass/Fail Criteria

- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`. Note that there is no acknowledgment for activating a `DefaultDERControl` in the IEEE 2030.5 protocol. Verification of the activation of the `DefaultDERControl` must be done out-of-band.
- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) to the Server.
- **[C]** Client aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration has elapsed
- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`. Note that there is no acknowledgment for activating a `DefaultDERControl` in the IEEE 2030.5 protocol. Verification of the activation of the `DefaultDERControl` must be done out-of-band.
- Client fails if EDB1 and/or EDB2 POSTs responses to the TFA event.

AGG-005 - Aggregator Event - 1 DERP, 1 DDERC, 2 Non-overlapping Similar DERC [A, S]

Purpose

The aggregator event (1 DERP, 1 DDERC, 2 non-overlapping similar DERC) test verifies that the aggregator executes two similar non-overlapping DER events with a `DefaultDERC` from one `DERProgram` on the correct end devices.

Setup

1. **[S]** Perform the test setup from AGG-001.
2. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the TFA `DERProgram`
3. **[S]** Create a `DERControl` for a specific control (for example, `opModFixedPFInjectW`) for the TFA `DERProgram` with a start time of two minutes from now and duration of one minute.
4. **[S]** Create a `DERControl` for the same specific control for the TFA `DERProgram` with a start time of four minutes from now with duration of one minute.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
3. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the event created in setup step 2 to the Server.
4. **[C]** aggregator applies the TFA `DERControl` created in setup step 2 at the correct start time for the correct duration.
5. **[C]** For EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the event created in setup step 2.
6. **[C]** For EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the event created in setup step 2 has elapsed.
7. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the event created in setup step 2.
8. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the event created in setup step 3 to the Server.
9. **[C]** aggregator applies the TFA `DERControl` created in setup step 3 at the correct start time for the correct duration.
10. **[C]** For EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the event created in setup step 3.
11. **[C]** For EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the event created in setup step 3 has elapsed.

12. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the event created in setup step 3.

Pass/Fail Criteria

- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
- **[C, S]** Client, f or EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the third event to the Server.
- **[C]** Client, aggregator applies the third TFA `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the third event.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the third event has elapsed.
- **[C]** Client, or EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the third TFA `DERControl`.
- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the second event to the Server.
- **[C]** Client, aggregator applies the second TFA `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the second event.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the second event has elapsed.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the second TFA `DERControl`.
- Client fails if EDB1 and/or EDB2 POSTs responses to the TFA event.

AGG-006 - Aggregator Event - 2 DERP, 2 DDERC, 2 Non-overlapping Similar DERC [A, S]

Purpose

The aggregator event (2 DERP, 2 DDERC, 2 non-overlapping similar DERC) test verifies that the aggregator executes two similar, non-overlapping DER events with a `DefaultDERC` from two `DERPrograms` on the correct end devices. The two `DERPrograms` have different primacy so the aggregator must use the `DefaultDERC` from the program with the lower primacy value (higher priority).

Setup

1. **[S]** Perform the test setup from AGG-001.
2. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`
3. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the TFA `DERProgram`.
4. **[S]** Create a `DERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram` with a start time of two minutes from now and duration of one minute.
5. **[S]** Create a `DERControl` for the same specific control for the SY `DERProgram` with a start time of four minutes from now with duration of one minute.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C, S]** Configure resources as outlined in Setup and wait for Client to acquire resources.
3. Wait seven minutes after acquiring resources before terminating the test.
4. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
5. **[C]** For EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram`.
6. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the event created in setup step 3 to the Server.
7. **[C]** aggregator applies the TFA `DERControl` created in setup step 3 at the correct start time for the correct duration.
8. **[C]** For EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the event created in setup step 3.
9. **[C]** For EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the event created in setup step 3 has elapsed.
10. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the event created in setup step 3.

11. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the event created in setup step 4 to the Server.
12. **[C]** aggregator applies the TFA `DERControl` created in setup step 4 at the correct start time for the correct duration.
13. **[C]** For EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the event created in setup step 4.
14. **[C]** For EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the event created in setup step 4 has elapsed.
15. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the event created in setup step 4.

Pass/Fail Criteria

- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
- **[C]** Client, for EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram`.
- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the third event to the Server.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the third TFA `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the third event.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the third event has elapsed.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the third TFA `DERControl`.
- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the second event to the Server.
- **[C]** Client, aggregator applies the second TFA `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the second event.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the second event has elapsed.
- **[S]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the second TFA `DERControl`.

AGG-007 - Aggregator Event - 2 DERP, 2 DDERC, 2 Overlapping Similar DERC - System DERC followed by Transformer DERC before Start of System DERC [A, S]

Purpose

The aggregator event (2 DERP, 2 DDERC, 2 overlapping similar DERC - system DERC followed by service point DERC before start of system DERC) test verifies event handling of two, overlapping events using the same DER control.

The higher priority TFA control overlaps with the previously scheduled SY control. Both events are scheduled ahead of time. Because the TFA control is higher priority and the aggregator should have discovered both events before the start of the events, the aggregator should execute only the TFA event.

Setup

1. **[S]** Perform the test setup from AGG-001.
2. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`
3. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the TFA `DERProgram`.
4. **[S]** Create a `DERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram` with a start time of two minutes from now and duration of two minute.
5. **[S]** Create a `DERControl` for the same specific control for the TFA `DERProgram` with a start time of three minutes from now with a one-minute duration.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
3. **[C]** For EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram`.
4. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator POSTs response with status 1 (Event Received) for the SY event to the Server.
5. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the TFA event to the Server.
6. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 7 (Event Superseded) for the SY event to the Server.
7. **[C]** For EDB1 and EDB2, aggregator applies the SY `DERControl` at the correct start time for the correct duration.
8. **[C]** For EDB1 and EDB2, POSTs response with status 2 (Event Started) at start time of the SY event.

9. **[C]** For EDA1 and EDA2, aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
10. **[C]** For EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the TFA event.
11. **[C]** For EDB1 and EDB2, POSTs response with status 3 (Event Completed) after duration of the SY event has elapsed.
12. **[C]** For EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` after the completion of the SY `DERControl`.
13. **[C]** For EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the TFA event has elapsed.
14. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`.

Pass/Fail Criteria

- **[C, S]** Configure resources as outlined in Setup and wait for Client to acquire resources.
- Wait seven minutes after acquiring resources before terminating the test.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
- **[C]** Client, for EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram`.
- **[C, S]** Client, for EDA1, EDA2, EDB1, and EDB2, aggregator POSTs response with status 1 (Event Received) for the SY event to the Server.
- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the TFA event to the Server.
- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 7 (Event Superseded) for the SY event to the Server.
- **[C]** Client, for EDB1 and EDB2, aggregator applies the SY `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client, for EDB1 and EDB2, POSTs response with status 2 (Event Started) at start time of the SY event.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the TFA event.
- **[C, S]** Client, for EDB1 and EDB2, POSTs response with status 3 (Event Completed) after duration of the SY event has elapsed.
- **[C]** Client, for EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` after the completion of the SY `DERControl`.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the TFA event has elapsed.

- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`.

AGG-008 - Aggregator Event - 2 DERP, 2 DDERC, 2 Overlapping Similar DERC - Transformer DERC followed by System DERC [A, S]

Purpose

The aggregator event (2 DERP, 2 DDERC, 2 overlapping similar DERC - service point DERC followed by system DERC) test verifies the event handling of two, overlapping events using the same DER control.

The lower priority SY control overlaps with the previously scheduled TFA control. Both events are scheduled ahead of time. Because the TFA control is higher priority and the aggregator should have discovered both evenets before the start of the events, the aggregator should execute only the TFA event.

Setup

1. **[S]** Perform the test setup from AGG-001.
2. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`.
3. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the TFA `DERProgram`.
4. **[S]** Create a `DERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram` with a start time of three minutes from now and a two-minute duration.
5. **[S]** Create a `DERControl` for the same specific control for the TFA `DERProgram` with a start time of two minutes from now and a two-minute duration.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
3. **[C]** For EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram`.
4. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator POSTs response with status 1 (Event Received) for the SY event to the Server.
5. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the TFA event to the Server.
6. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 7 (Event Superseded) for the SY event to the Server.
7. **[C]** For EDA1 and EDA2, aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
8. **[C]** For EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the TFA event.

9. **[C]** For EDB1 and EDB2, aggregator applies the SY `DERControl` at the correct start time for the correct duration.
10. **[C]** For EDB1 and EDB2, POSTs response with status 2 (Event Started) at start time of the SY event.
11. **[C]** For EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the TFA event has elapsed.
12. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`.
13. **[C]** For EDB1 and EDB2, POSTs response with status 3 (Event Completed) after duration of the SY event has elapsed.
14. **[C]** For EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` after the completion of the SY `DERControl`.

Pass/Fail Criteria

- **[C]** Client, Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
- **[C]** Client, for EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram`.
- **[C, S]** Client, for EDA1, EDA2, EDB1, and EDB2, aggregator POSTs response with status 1 (Event Received) for the SY event to the Server.
- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the TFA event to the Server.
- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 7 (Event Superseded) for the SY event to the Server.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the TFA event.
- **[C]** Client, for EDB1 and EDB2, aggregator applies the SY `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client, for EDB1 and EDB2, POSTs response with status 2 (Event Started) at start time of the SY event.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the TFA event has elapsed.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`.
- **[C, S]** Client, for EDB1 and EDB2, POSTs response with status 3 (Event Completed) after duration of the SY event has elapsed.
- **[C]** Client, for EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` after the completion of the SY `DERControl`.

AGG-009 - Aggregator Event - 2 DERP, 2 DDERC, 2 Overlapping Similar DERC - Transformer DERC followed by System DERC after Start of System Event [A, S]

Purpose

The aggregator event (2 DERP, 2 DDERC, 2 overlapping similar DERC - service point DERC followed by system DERC after start of system event) test verifies event handling of two, overlapping events using the same DER control.

The higher priority TFA control overlaps with the previously scheduled SY control. The TFA control is scheduled after the start of the SY control. Because the TFA control is higher priority, the aggregator should execute the SY control until the start time of the TFA control followed by TFA control execution.

Setup

1. **[S]** Perform the test setup from AGG-001.
2. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`.
3. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the TFA `DERProgram`.
4. **[S]** Create a `DERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram` with a start time of one minute from now and a four-minute duration.
5. **[S]** After the start of the SY event, create a `DERControl` for the same specific control for the TFA `DERProgram` with a start time of two minutes from now and a two-minute duration.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
3. **[C]** For EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram`.
4. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator POSTs response with status 1 (Event Received) for the SY event to the Server.
5. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator applies the SY `DERControl` at the correct start time for the correct duration.
6. **[C]** EDA1, EDA2, EDB1, and EDB2, POSTs response with status 2 (Event Started) at start time of the SY event.
7. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the TFA event to the Server.

8. **[C]** For EDA1 and EDA2, aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
9. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 7 (Event Superseded) for the SY event to the Server.
10. **[C]** For EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the TFA event.
11. **[C]** For EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the TFA event has elapsed.
12. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`.
13. **[C]** For EDB1 and EDB2, POSTs response with status 3 (Event Completed) after duration of the SY event has elapsed.
14. **[C]** For EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` after the completion of the SY `DERControl`.

Pass/Fail Criteria

- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
- **[C]** Client, for EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram`.
- **[C, S]** Client, for EDA1, EDA2, EDB1, and EDB2, aggregator POSTs response with status 1 (Event Received) for the SY event to the Server.
- **[C, S]** Client, for EDA1, EDA2, EDB1, and EDB2, aggregator applies the SY `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client EDA1, EDA2, EDB1, and EDB2, POSTs response with status 2 (Event Started) at start time of the SY event.
- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the TFA event to the Server.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 7 (Event Superseded) for the SY event to the Server.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the TFA event.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the TFA event has elapsed.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`.
- **[C, S]** Client, for EDB1 and EDB2, POSTs response with status 3 (Event Completed) after duration of the SY event has elapsed.

- **[C]** Client, for EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` after the completion of the SY `DERControl`.

AGG-010 - Aggregator Event - 2 DERP, 2 DDERC, 2 Overlapping Independent DERC - System DERC followed by Transformer DERC before Start of System DERC [A, S]

Purpose

The aggregator event (2 DERP, 2 DDERC, 2 overlapping independent DERC - SY DERC followed by TFA DERC before start of system DERC) test verifies event handling of two, overlapping events using the independent DER controls. Because the controls are independent, the aggregator should execute both controls.

Note: With the change in S1 to make `DERControl` independent, `CurrentDERProgram` no longer has meaning. Errata will be added to the S1 comments and addressed in the next IEEE 2030.5 Technical Working Group. For now, do not use the `CurrentDERProgram` resource.

Setup

1. **[S]** Perform the test setup from AGG-001.
2. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`.
3. **[S]** Create a `DefaultDERControl` for a different control (for example, `opModFixedW`) for the TFA `DERProgram`.
4. **[S]** Create a `DERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram` with a start time of two minutes from now and a two-minute duration.
5. **[S]** Create a `DERControl` for the different (for example, `opModFixedW`) control for the TFA `DERProgram` with a start time of three minutes from now and a one-minute duration.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` prior to the start of the SY `DERControl`.
3. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
4. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator POSTs response with status 1 (Event Received) for the SY event to the Server.
5. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator applies the SY `DERControl` at the correct start time for the correct duration.
6. **[C]** EDA1, EDA2, EDB1, and EDB2, POSTs response with status 2 (Event Started) at start time of the SY event.
7. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the TFA event to the Server.

8. **[C]** For EDA1 and EDA2, aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
9. [C] For EDA1 and EDA2, POSTs response with status 2 (Event Started) at the start time of the TFA event.
10. **[C]** For EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the TFA event has elapsed.
11. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`.
12. **[C]** For EDA1, EDA2, EDB1 and EDB2, POSTs response with status 3 (Event Completed) after duration of the SY event has elapsed.
13. **[C]** For EDA1, EDA2, EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` after the completion of the SY `DERControl`.

Pass/Fail Criteria

- **[C]** Client, for EDA1, EDA2, EDB1, and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` prior to the start of the SY `DERControl`.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
- **[C, S]** Client, for EDA1, EDA2, EDB1, and EDB2, aggregator POSTs response with status 1 (Event Received) for the SY event to the Server.
- **[C]** Client, for EDA1, EDA2, EDB1, and EDB2, aggregator applies the SY `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client EDA1, EDA2, EDB1, and EDB2, POSTs response with status 2 (Event Started) at start time of the SY event.
- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the TFA event to the Server.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client EDA1 and EDA2 POSTs response with status 2 (Event Started) at start time of the TFA event.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the TFA event has elapsed.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`.
- **[C, S]** Client, for EDA1, EDA2, EDB1 and EDB2, POSTs response with status 3 (Event Completed) after duration of the SY event has elapsed.
- **[C]** Client, for EDA1, EDA2, EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` after the completion of the SY `DERControl`.

AGG-011 - Aggregator Event - 2 DERP, 2 DDERC, 2 Overlapping Independent DERC - Transformer DERC followed by System DERC [A, S]

Purpose

The aggregator event (2 DERP, 2 DDERC, 2 overlapping independent DERC - service point DERC followed by system DERC) test verifies event handling of two, overlapping events using independent DER controls. Because the controls are independent, the aggregator should execute both controls.

Note: With the change in S1 to make `DERControl` independent, `CurrentDERProgram` no longer has meaning. Errata will be added to the S1 comments and addressed in the next IEEE 2030.5 Technical Working Group. For now, do not use the `CurrentDERProgram` resource.

Setup

1. **[S]** Perform the test setup from AGG-001.
2. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`.
3. **[S]** Create a `DefaultDERControl` for a different control (for example, `opModFixedW`) for the TFA `DERProgram`.
4. **[S]** Create a `DERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram` with a start time of three minutes from now and a two-minute duration.
5. **[S]** Create a `DERControl` for the different (for example, `opModFixedW`) control for the TFA `DERProgram` with a start time of two minutes from now and a two-minute duration.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` prior to the start of the SY `DERControl`.
3. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
4. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the TFA event to the Server.
5. **[C]** For EDA1 and EDA2, aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
6. **[C]** EDA1 and EDA2 POSTs response with status 2 (Event Started) at start time of the TFA event.
7. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator POSTs response with status 1 (Event Received) for the SY event to the Server.
8. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator applies the SY `DERControl` at the correct start time for the correct duration.

9. **[C]** EDA1, EDA2, EDB1, and EDB2, POSTs response with status 2 (Event Started) at start time of the SY event.
10. **[C]** For EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the TFA event has elapsed.
11. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`.
12. **[C]** For EDA1, EDA2, EDB1 and EDB2, POSTs response with status 3 (Event Completed) after duration of the SY event has elapsed.
13. **[C]** For EDA1, EDA2, EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` after the completion of the SY `DERControl`.

Pass/Fail Criteria

- **[C]** Client, for EDA1, EDA2, EDB1, and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` prior to the start of the SY `DERControl`.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the TFA event to the Server.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client EDA1 and EDA2 POSTs response with status 2 (Event Started) at start time of the TFA event.
- **[C, S]** Client, for EDA1, EDA2, EDB1, and EDB2, aggregator POSTs response with status 1 (Event Received) for the SY event to the Server.
- **[C]** Client, for EDA1, EDA2, EDB1, and EDB2, aggregator applies the SY `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client EDA1, EDA2, EDB1, and EDB2, POSTs response with status 2 (Event Started) at start time of the SY event.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the TFA event has elapsed.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`.
- **[C, S]** Client, for EDA1, EDA2, EDB1 and EDB2, POSTs response with status 3 (Event Completed) after duration of the SY event has elapsed.
- **[C]** Client, for EDA1, EDA2, EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` after the completion of the SY `DERControl`.

AGG-012 - Aggregator Event - 2 DERP, 2 DDERC, 2 Overlapping Independent DERC - Transformer DERC followed by System DERC after Start of System Event [A, S]

Purpose

The aggregator event (2 DERP, 2 DDERC, 2 overlapping independent DERC - transformer DERC followed by system DERC after start of system event) test verifies event handling of two, overlapping events using independent DER controls. Because the controls are independent, the aggregator should execute both controls.

Note: With the change in S1 to make `DERControl` independent, `CurrentDERProgram` no longer has meaning. Errata will be added to the S1 comments and addressed in the next IEEE 2030.5 Technical Working Group. For now, do not use the `CurrentDERProgram` resource.

Setup

1. **[S]** Perform the test setup from AGG-001.
2. **[S]** Create a `DefaultDERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram`.
3. **[S]** Create a `DefaultDERControl` for a different control (for example, `opModFixedW`) for the TFA `DERProgram`.
4. **[S]** Create a `DERControl` for a specific control (for example, `opModFixedPFInjectW`) for the SY `DERProgram` with a start time of one minute from now and a four-minute duration.
5. **[S]** After the start of the SY event, create a `DERControl` for the different control (for example, `opModFixedW`) for the TFA `DERProgram` with a start time of two minutes from now and a two-minute duration.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
3. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram`.
4. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator POSTs response with status 1 (Event Received) for the SY event to the Server.
5. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator applies the SY `DERControl` at the correct start time for the correct duration.
6. **[C]** EDA1, EDA2, EDB1, and EDB2, POSTs response with status 2 (Event Started) at start time of the SY event.
7. **[C]** For EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the TFA event to the Server.

8. **[C]** For EDA1 and EDA2, aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
9. **[C]** For EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the TFA event.
10. **[C]** For EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the TFA event has elapsed.
11. **[C]** For EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`.
12. **[C]** For EDA1, EDA2, EDB1 and EDB2, POSTs response with status 3 (Event Completed) after duration of the SY event has elapsed.
13. **[C]** For EDA1, EDA2, EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` after the completion of the SY `DERControl`.

Pass/Fail Criteria

- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` prior to the start of the TFA `DERControl`.
- **[C]** Client, for EDA1, EDA2, EDB1, and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram`.
- **[C, S]** Client, for EDA1, EDA2, EDB1, and EDB2, aggregator POSTs response with status 1 (Event Received) for the SY event to the Server.
- **[C]** Client, for EDA1, EDA2, EDB1, and EDB2, aggregator applies the SY `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client EDA1, EDA2, EDB1, and EDB2, POSTs response with status 2 (Event Started) at start time of the SY event.
- **[C, S]** Client, for EDA1 and EDA2, aggregator POSTs response with status 1 (Event Received) for the TFA event to the Server.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the TFA `DERControl` at the correct start time for the correct duration.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 2 (Event Started) at start time of the TFA event.
- **[C, S]** Client, for EDA1 and EDA2, POSTs response with status 3 (Event Completed) after duration of the TFA event has elapsed.
- **[C]** Client, for EDA1 and EDA2, aggregator applies the `DefaultDERControl` for the TFA `DERProgram` after the completion of the TFA `DERControl`.
- **[C, S]** Client, for EDA1, EDA2, EDB1 and EDB2, POSTs response with status 3 (Event Completed) after duration of the SY event has elapsed.
- **[C]** Client, for EDA1, EDA2, EDB1 and EDB2, aggregator applies the `DefaultDERControl` for the SY `DERProgram` after the completion of the SY `DERControl`.

10 Error Handling Tests

This test section covers the 5.8 Error Handling section of the CSIP guide. These tests target the DER client ability to handle and report operational errors, as required by the CSIP guide. These tests include:

- HTTP error handling
- subscription termination handling

Before running these tests, the DER client shall have previously completed the tests described in Communication Fundamentals Tests, Core Function Set Tests, and Basic Functions Tests of this document.

ERR-001 - Error Scenario 1 [C, A]

Purpose

The error scenario 1 test verifies error handling for various IEEE 2030.5 server and client scenarios. This test shall include scenarios where certain HTTP response codes are used by a server.

Setup

1. **[TS]** Verify a `DeviceCapability` resource exists on the Server and include at least one resource in the `DeviceCapability` resource. The `DeviceCapability` is available only through TLS port and if an unencrypted HTTP GET is received, the Server shall respond with an HTTP `301`, `Moved Permanently`, or HTTP `302`, `Redirect`, which include a location header to the new URI of the requested resource.

Procedure

1. **[TS]** Record the Client/Server communications.
2. **[C]** Do an HTTP GET on `DeviceCapability` resource from the Server using the supported HTTP and IP address.
3. **[TS]** Respond to the HTTP GET request by responding with either HTTP `301 Moved Permanently` or HTTP `302 Redirect` and Location header specifying the new URI of the resource being requested.
4. **[C]** Process the HTTP `301` or `302` messages, including the location header and prepare a new HTTP request and perform an HTTP GET request using the TLS port using the URI included in the response.
5. **[TS]** Respond to the HTTP GET request by responding with the resource body of the `DeviceCapability` resource and HTTP `200 OK` response.
6. **[C]** Process the retrieved `DeviceCapability` resource and verify there is at least one resource in the `DeviceCapability`.
7. **[C]** Perform an HTTP GET operation on the found resource in step 2 and process the response payload from the Server.

Pass/Fail Criteria

- **[C]** The Client successfully requested the `DeviceCapability` resource from the Server using the HTTP configuration information provided.
- **[TS]** Server responded with HTTP `301` or `302` and Location header for the new URI.
- **[C]** Client received the HTTP `301` or `302` responses, processed the Location header which was provided in the HTTP response and successfully did an HTTP GET using the TLS port and URI.
- **[TS, C]** Server successfully received the HTTP GET request on the TLS port and URI and responded with HTTP `200 OK` with `DeviceCapability` payload.

- **[TS, C]** Client found at least one resource included in the `DeviceCapability`. Server included at least one resource link in the returned `DeviceCapability` to the Client.
- **[TS, C]** The Client successfully received the payload of the found resource from the `DeviceCapability` resource. The Server responded with `200 OK` and returned a conformant payload for its resource.

ERR-002 - Error Scenario 2 [A, S]

Purpose

The error scenario 2 test verifies the error handling of various IEEE 2030.5 server and client scenarios. This test shall include client subscription to a resource and, subsequently, the subscription is canceled by a server, which should trigger the client to post another subscription request. This test shall also include the server sending an undesired subscription to a client, which shall return an HTTP `400` error and the server shall delete the subscription without notification.

Setup

1. Server shall be able to power reset itself and preserve the subscription requests
2. Server shall be able to cancel an outstanding subscription and send cancellation Notification to clients.
3. Server shall be able to send invalid Notification to clients.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[S]** Power reset the system, so it restarts. All outstanding subscriptions shall be persistent across a power reset. After a successful power reset, cause a change on one of the resources in the subscribed `FunctionSetAssignmentsList` which shall trigger a notification message to be sent to the Client.
3. **[C]** When the Notification message is received from the Server, process the incoming Notification message and payload, including validation and respond to the Notification message with HTTP `201 Created` or `204 No Content` message.
4. **[C]** Perform an HTTP GET request on the href of the resource included in the Server Notification message. Process the payload returned from the Server from the HTTP GET request and compare to the Notification body payload. They shall be identical.
5. **[TS]** Cancel the Client subscription request by sending a notification to the URI provided in the original subscription with the following values:
    - Notification/`status` = `1` (Subscription canceled, no additional information).
    - Notification/`subscriptionURI` = URI of the subscription instance created by the server for the original request.
6. **[C]** Process the incoming Notification cancellation message from the Server by removing it from own list of outstanding Subscriptions.
7. **[TS]** Prepare and send an invalid Notification message to the outstanding Client `FunctionSetAssignmentsList` subscription by using the following values:
    - Instead of the `FunctionSetAssignmentsList` items to be included in the body of the notification message, insert `EndDevice` instance information.
8. **[C]** Receiving an invalid Notification sent by the Server shall cause the client to respond with an HTTP `400` error.

9. **[S]** On receipt of the HTTP `400` error, the subscription for which the notification was sent in step 8 shall be deleted without notification.

Pass/Fail Criteria

- **[S]** Server was able to power reset the system, so it restarted. All outstanding subscriptions shall be persistent across a power reset. After a successful power reset, Server caused a change on one of the resources in the subscribed `FunctionSetAssignmentsList` and sent Notification message to the Client with the right Notification body as requested by the original subscription request. For example, limit.
- **[S, C]** The Client successfully received and processed the incoming Notification message and payload and validated and responded back to the Notification message with HTTP `201 Created` or HTTP `204 No Content` message.
- **[S, C]** The Client successfully did an HTTP GET request on the href of the resource included in the Server Notification message. Successfully processed the payload returned from the Server from the HTTP GET request and validated the Notification body payload. They are identical.
- **[TS, C]** Server canceled the Client subscription request by sending a notification to the URI provided in the original subscription with the following values:
    - Notification/`status` = `1` (Subscription canceled, no additional information)
    - Notification/`subscriptionURI` = URI of the subscription instance created by the server for the original request
- **[C]** The Client successfully received and processed the incoming Notification cancellation message from the Server and deleted it from own list of outstanding Subscriptions.
- **[TS, C]** Server successfully prepared and sent an invalid Notification message to the outstanding Client `FunctionSetAssignmentsList` subscription by using the following values:

    Instead of the `FunctionSetAssignmentsList` items to be included in the body of the notification message, used `EndDevice` instance information.

- **[TS, C]** The Client successfully received an invalid Notification sent by the Server and responded with an HTTP `400` error.
- **[TS, C]** On receipt of the HTTP `400` error, Server deleted the associated subscription and did so without sending the Client a notification.

11 Maintenance of the Model Tests

This test section covers the 5.9 Maintenance of the Model section of the CSIP guide. These tests target the utility server ability to manage CSIP model changes and the DER client ability to react to the changing conditions. These tests include:

- inverter maintenance
- group maintenance
- controls maintenance
- program maintenance
- subscription maintenance

Bofore running these tests, the DER client and server EUTs shall have previously completed the tests described in the Communication Fundamentals Tests, Core Function Set Tests, and Basic Functions Tests sections of this document.

For these tests, the aggregator is assumed to be managing four `EndDevices` (EDA1, EDA2, EDB1, and EDB2) as shown in the following topology:

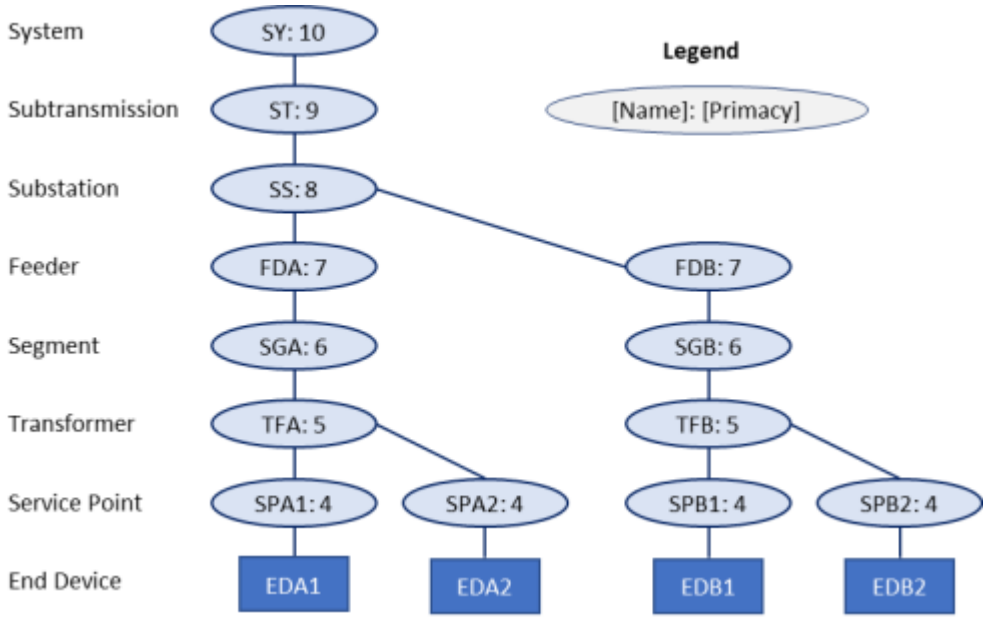

Figure 16 Aggregator End Device Topology for Maintenance Tests

IEEE 2030.5 permits many ways to create this topology using `FunctionSetAssignments` and associated `DERProgram`, but there must be a `DERProgram` associated with each group node used to send DER controls to all devices connected to that node.

MAINT-001 - Inverter Maintenance (Out-Of-Band) [A, S]

Purpose

The out-of-band (OOB) inverter maintenance test verifies inverter population change handling. The inverter population is managed by an aggregator using an OOB method. This test shall test scenarios that include inverter addition and deletion using `EndDevice`/`EndDeviceList` for a client device/aggregator.

Setup

1. Follow test setup section in the AGG-001 - Aggregator Operation Subscription test with additional setup as described in this section. Client in this test represents an aggregator, which manages number of individual DERs in its `EndDeviceList`. At the completion of the AGG-001 - Aggregator Operation Subscription test, the set of `EndDevice` instances included for the aggregator client are:
    - SPA1: EDA1 and EDA1X
    - SPA2: EDA2
    - SPB1: EDB1
    - SPB2: EDB2
2. Server shall configure the `EndDeviceList` and each of its `EndDevice` instances for the Client with an attribute that can be subscribed to.
3. Client shall be able to subscribe to its `EndDeviceList` and receive notifications for changes when they occur.
4. Server shall be able to send notifications to subscribed clients when the subscribed resource changes.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Subscribe to the `EndDeviceList` so notifications can be received when there are changes to each instance and to the list.
3. The aggregator and Utility agreed out-of-band that EDA1X is no longer being managed by the aggregator and needs to be deleted from its list.
4. **[S]** Delete the EDA1X `EndDevice` instance from the aggregator `EndDeviceList` and send a notification to the `EndDeviceList` subscriptions.
5. **[C]** Receive the notification for the `EndDeviceList` and do additional HTTP GETs to receive the rest of the `EndDeviceList` contents not sent as part of the Notification message payload. Client shall also receive a notification on the deleted `EndDevice` instance. Based on the received `EndDeviceList`, it shall notice the EDA1X `EndDevice` instance is deleted and its internal state should also reflect the change so it no longer manages that device.
6. **[TC]** Do a GET on the EDA1X `EndDevice` href and it shall result in an HTTP `404 Not Found` error.

Pass/Fail Criteria

- **[C, S]** The Client successfully subscribed to the `EndDevice` instances and `EndDeviceList` and received an HTTP `201 Created` response.
- The aggregator and Utility successfully agreed that EDA1X is no longer managed by the aggregator and its `EndDevice` instance will be deleted by the Server.
- **[C, S]** Server successfully deleted the EDA1X `EndDevice` instance from the aggregator `EndDeviceList` and sent a notification to the `EndDeviceList` subscriptions. The notification resource body for the `EndDeviceList` was correctly formed using the `limit` parameter requested by the Client in the original subscription request.
- **[C, S]** The Client successfully received the notification for the `EndDeviceList` and did additional HTTP GETs to receive the rest of the `EndDeviceList` contents not sent as part of the Notification message payload. Client also received a notification on the deleted `EndDevice` instance. Based on the received `EndDeviceList`, Client found the EDA1X `EndDevice` instance is deleted and its internal state was updated so it no longer manages that device.
- **[TC, S]** The Client successfully did an HTTP GET on the EDA1X `EndDevice` href and received an HTTP `404 Not Found` error.

MAINT-002 - Inverter Maintenance (In-Band) [A, S]

Purpose

The in-band inverter maintenance test verifies inverter population change handling. The inverter population is managed by an aggregator using an in-band method. This test shall test scenarios that include inverter addition and deletion using `EndDevice`/`EndDeviceList` for a client device/aggregator.

Setup

1. Follow the test setup section in the AGG-001 - Aggregator Operation Subscription test with additional setup as described in this section. Client in this test represents an aggregator, which manages number of individual DERs in its `EndDeviceList`. At the completion of the AGG-001 - Aggregator Operation Subscription test, the set of `EndDevice` instances included for the aggregator client are:
    - SPA1: EDA1 and EDA1X
    - SPA2: EDA2
    - SPB1: EDB1
    - SPB2: EDB2
2. Server shall configure the `EndDeviceList` and each of its `EndDevice` instances for the Client with an attribute that can be subscribed to.
3. Client shall be able to subscribe to its `EndDeviceList` and receive notifications for changes when they occur.
4. Server shall be able to send notifications to subscribed clients when the subscribed resource changes.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Subscribe to the EndDeviceList so notifications can be received when there are changes to the list.
3. Aggregator found that EDA1X is no longer being managed by the aggregator and needs to be deleted from its list.
4. **[C]** Do an HTTP DELETE operation on the EDA1X `EndDevice` instance href to delete it from the aggregator `EndDeviceList`.
5. **[S]** Receive and process the HTTP DELETE operation on the EDA1X `EndDevice` instance which shall also trigger Notification messages on the `EndDeviceList` for the EDA1X `EndDevice` instance.
6. **[C]** Receive the notification for the `EndDeviceList` and do additional HTTP GETs to receive the rest of the `EndDeviceList` contents not sent as part of the Notification message payload. Client shall also receive a notification on the deleted `EndDevice` instance. Based on the received `EndDeviceList`, it shall notice the EDA1X

EndDevice instance is deleted and its internal state should also reflect the change so it no longer manages that device.

7. **[C]** Perform an HTTP GET operation on the EDA1X EndDevice href and it shall result in an HTTP 404 Not Found error.

Pass/Fail Criteria

- **[C, S]** The Client successfully subscribed to the EndDevice instances and EndDeviceList and received an HTTP 201 Created response.
- **[C, S]** The aggregator found that EDA1X is no longer managed by the aggregator and its EndDevice instance will be deleted.
- **[C, S]** Server successfully received the HTTP DELETE message and deleted the EDA1X EndDevice instance from the aggregator EndDeviceList and sent a notification to the EndDeviceList for the EDA1X EndDevice subscriptions. The notification resource body for the EndDeviceList was correctly formed using the limit parameter requested by the Client in the original subscription request.
- **[C, S]** The Client successfully received the notification for the EndDeviceList and did additional HTTP GETs to receive the rest of the EndDeviceList contents not sent as part of the Notification message payload. Client also received a notification for the deleted EndDevice instance. Based on the received EndDeviceList, Client found the EDA1X EndDevice instance is deleted and its internal state was updated so it no longer manages that device.
- **[C, S]** The Client successfully did an HTTP GET on the EDA1X EndDevice href and received an HTTP 404 Not Found error.

MAINT-003 - Group Maintenance [A, S]

Purpose

The group maintenance test verifies topology change handling. The topology is managed by the utility server using `FunctionSetAssignments`. This test shall test scenarios that include topology changes that cause changes in the associated `FunctionSetAssignment` groups.

Setup

1. Follow test setup section in the AGG-001 - Aggregator Operation Subscription test with additional setup as described in this section. Client in this test represents an aggregator, which manages number of individual DERs in its `EndDeviceList`. At the completion of the AGG-001 - Aggregator Operation Subscription test, the set of `EndDevice` instances included for the aggregator client are:
    - SPA1: EDA1
    - SPA2: EDA2
    - SPB1: EDB1
    - SPB2: EDB2
2. Server shall configure the `FunctionSetAssignmentsList` for each of its `EndDevice` instances for the Client with an attribute that can be subscribed to.
3. Client shall be able to subscribe to its `FunctionSetAssignmentsList` for each of the `EndDevice` instance that it manages and receive notifications for changes when they occur.
4. Server shall be able to send notifications to subscribed clients when the subscribed resource changes.
5. Server shall create `DERProgram` for TFA and TFB nodes respectively.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** For EDA1, EDA2, EDB1, and EDB2, aggregator subscribes to the `FunctionSetAssignmentsList` of each `EndDevice`.
3. **[C]** For EDA1 and EDA2, applies the TFA `DefaultDERControl`.
4. **[C]** For EDB1 and EDB2, applies the TFB `DefaultDERControl`.
5. **[S]** Moves the EDB1 from the SPB1 node to the SPA1 node. This changes the group membership of EDB1. It causes the EDB1 `FunctionSetAssignmentsList`/`DERProgramList` to change.
6. **[S]** Sends notification to aggregator that EDB1 `FunctionSetAssignmentsList` has changed.
7. **[C]** aggregator receives the notification that EDB1 `FunctionSetAssignmentsList` has changed.
8. **[C]** aggregator GETs the new EDB1 `FunctionSetAssignmentsList` and the `DERProgramList` to which that EDB1 now belongs.

9. **[C]** If needed, aggregator cancels the subscriptions to the old `FunctionSetAssignmentsList` for EDB1.
10. **[C]** If needed, aggregator subscribes to the new `FunctionSetAssignmentsList` of EDB1.

Pass/Fail Criteria

- **[C, S]** Aggregator (Client) successfully subscribed to each of the `DERProgramList` of each end device: EDA1, EDA2, EDB1, and EDB2. Server responded successfully to the Subscription requests and sent back correct responses to the Aggregator.
- **[C, S]** Aggregator successfully subscribed to the `FunctionSetAssignmentsList` for each of the `EndDevice` instances: EDA1, EDA2, EDB1, and EDB2. Server responded successfully to the Subscription requests and sent back correct responses to the Aggregator.
- **[C, S]** EDA1 and EDA2 end devices correctly processed the `DERProgramList` associated with its device and applied the TFA `DefaultDERControl`.
- **[C, S]** EDB1 and EDB2 end devices correctly processed the `DERProgramList` associated with its device and applied the TFB `DefaultDERControl`.
- **[C, S]** Server successfully moved the EDB1 from the SPB1 node to the SPA1 node that caused the EDB1 `FunctionSetAssignmentsList/DERProgramList` to change.
- **[C, S]** Server successfully sent a notification to aggregator that EDB1 `FunctionSetAssignmentsList` has changed.
- **[C, S]** Aggregator successfully received the notification that EDB1 `FunctionSetAssignmentsList` has changed and processes the message in order to determine that it needed to retrieve the updated DERProgramList information.
- **[C, S]** Aggregator successfully performed GETs the new EDB1 `FunctionSetAssignmentsList` and all `DERPrograms`, to which that EDB1 now belongs. Server responded successfully to each of the GETs from the Aggregator and sent correct payload as requested.
- **[C, S]** If needed, the aggregator has canceled any subscriptions to the old `FunctionSetAssignmentsList` for EDB1.
- **[C, S]** If needed, the aggregator has successfully subscribed to the new `FunctionSetAssignmentsList` of EDB1.

MAINT-004 - Maintenance of Controls [A, S]

Purpose

The maintenance of controls test verifies inverter DER control change handling. The changes are issued by the Utility Server. This test shall test scenarios that include `DERControlList`/`DERControl` changes assigned to the aggregator/client device.

Setup

1. Follow the test setup section in the AGG-001 - Aggregator Operation Subscription test with additional setup as described in this section. Client in this test represents an aggregator, which manages number of individual DERs in its `EndDeviceList`. At the completion of the AGG-001 - Aggregator Operation Subscription test, the set of `EndDevice` instances included for the aggregator client are:
    - SPA1: EDA1
    - SPA2: EDA2
    - SPB1: EDB1
    - SPB2: EDB2
2. Server shall configure the `DERProgramList` and `DERControlList` for each `EndDevice FunctionSetAssignments` with an attribute that can be subscribed to.
3. Client shall be able to subscribe to its `DERProgramList` and `DERControlList` and receive notifications for changes when they occur.
4. Server shall be able to add or change a `DERControl` to an existing `DERControlList` and send Notification each time a change occurs.
5. Server shall create `DERProgram` for TFA and TFB nodes respectively (equal primacy).

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Subscribe to the `DERProgramList` and `DERControlList` for each `EndDevice FunctionSetAssignments` instance so notifications can be received when there are changes to each instance. Use the `all` value of the `DERControlList` as the `limit` parameter to the subscription request.
3. **[S]** Add a new `DERControl` that starts in plus-fifteen minutes with duration of five minutes (no randomization) with at least one valid immediate or curve-based control to the TFA `DERProgram DERControlList`. Send a notification message for the affected `DERControlList`.
4. **[C]** Receive and process the Notification for the aggregator affected DERs. Client shall do additional HTTP GET on subordinate resources for the `DERControlList`.
5. **[C]** Apply the Event Processing rules to all `DERControls` for each `EndDevice`, including the newly created one from step 3 and schedule the `DERControl` event(s). Send a response for each `DERControl` if required by the Server.

Pass/Fail Criteria

- **[C, S]** The Client successfully subscribed to the `DERProgramList` and `DERControlList` for each `EndDevice FunctionSetAssignments` instance so notifications can be received when there are changes to each instance.
- **[C, S]** Server successfully added new `DERControl` which is scheduled to start in plus-fifteen minutes with duration of five minutes (no randomization) with at least one valid immediate or curve-based control to the TFA `DERProgram DERControlList`. Sent a notification message for the affected `DERControlList`.
- **[C, S]** The Client successfully received and processed the Notification for the affected aggregator DERs and did additional HTTP GETs on the resource the Notification message was sent for to receive the full content of the `DERControlList`. The Client successfully did additional HTTP GET on subordinate resources for the `DERControlList`.
- **[C, S]** Client applied the Event Processing rules to all `DERControls` for each `EndDevice`, including the newly created one from step 3 and scheduled the `DERControl` event(s). Sent a response for each `DERControl` if required by the Server.

MAINT-005 - Maintenance of Programs [A, S]

Purpose

The maintenance of programs test verifies inverter DER program change handling. DER program changes are issued by the utility server. This test shall test scenarios that include `DERProgramList`/`DERProgram` changes assigned to the aggregator/client device.

Setup

1. Follow the test setup section in the AGG-001 - Aggregator Operation Subscription test with additional setup as described in this section. Client in this test represents an aggregator, which manages number of individual DERs in its `EndDeviceList`. At the completion of the AGG-001 - Aggregator Operation Subscription test, the set of `EndDevice` instances included for the aggregator client are:
    - SPA1: EDA1
    - SPA2: EDA2
    - SPB1: EDB1
    - SPB2: EDB2
2. Server shall configure the `DERProgramList` and `DERControlList` for each `EndDevice FunctionSetAssignments` with an attribute that can be subscribed to.
3. Client shall be able to subscribe to its `DERProgramList` and `DERControlList` and receive notifications for changes when they occur.
4. Server shall be able to add or change a `DERProgram` with different primacy value and send Notification each time a change occurs.
5. Server shall create `DERProgram` (primacy=1) for TFA and TFB nodes and `DERProgram` (primacy=2) for SGA and SGB nodes respectively. TFA/TFB `DERProgram` shall have different DefaultDERControl values than SGA/SGB DERPrograms.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Subscribe to the `DERProgram` and `DERProgramList` for each `EndDevice FunctionSetAssignments` instance so notifications can be received when there are changes to each instance. Use the `all` value of the `DERProgramList` as the `limit` parameter to the subscription request.
3. **[S]** Swap the primacy values of TFA/TFB DERPrograms with the values of SGA/SGB DERPrograms. Send Notification messages to the affected `DERProgram` and `DERProgramList` with active subscriptions.
4. **[C]** Receive the incoming Notifications and process the resource information to reflect changes in the `DERProgram` instances, including Primacy value changes.
5. **[C]** Based on the updated Primacy values, activate the `DefaultDERControl` that belongs to the highest priority `DERProgram` if there is no active `DERControl` event.

Pass/Fail Criteria

- **[C, S]** The Client successfully subscribed to the `DERProgram` and `DERProgramList` for each `EndDevice FunctionSetAssignments` instance so notifications can be received when there are changes to each instance. Subscription request used the `all` value of the `DERProgramList` as the `limit` parameter.
- **[C, S]** Server found the two highest priority `DERPrograms` and swapped the values so the second highest priority `DERProgram` now becomes the highest. Sent Notification messages to the affected `DERProgram` and `DERProgramList` with active subscriptions.
- **[C]** The Client successfully received the incoming Notifications and processed the resource information to reflect changes in the `DERProgram` instances, including Primacy value changes.
- **[C]** Client, based on the updated Primacy values, activated the `DefaultDERControl` that belonged to the highest priority `DERProgram` if there is no active `DERControl` event.

MAINT-006 - Maintenance of Subscriptions [A, S]

Purpose

The maintenance of subscriptions test verifies subscription maintenance hosted by and subscribed to by end devices. This test shall test scenarios that include periodic aggregator client subscription renewal and shall test the polling fallback mechanism when subscriptions are not functioning.

Setup

1. Follow the test Setup section in the AGG-001 - Aggregator Operation Subscription test with additional setup as described in this section. Client in this test represents an aggregator, which manages number of individual DERs in its `EndDeviceList`. At the completion of the AGG-001 - Aggregator Operation Subscription test, the set of `EndDevice` instances included for the aggregator client are:
    - SPA1: EDA1
    - SPA2: EDA2
    - SPB1: EDB1
    - SPB2: EDB2
2. Server shall configure the `EndDevice` and `EndDeviceList` for the aggregator Client and its DER devices to be able to be subscribed to.
3. Client shall be able to subscribe to its `EndDeviceList` and its individual `EndDevice` entities and receive notifications for changes when they occur.
4. Server shall be able to add or change an `EndDeviceList` for the aggregator Client and receive subscription renewals.

Procedure

1. **[T]** Record the Client/Server communications.
2. **[C]** Subscribe to the EndDeviceList for the Aggregator so notifications can be received when there are changes to the EndDeviceList. Use the all value of the EndDeviceList as the limit parameter to the subscription request.
3. **[S]** Terminate the `EndDeviceList` subscription by sending a notification message with Subscription resource indicating status=1 (canceled, no additional information).
4. **[C]** Receive the Subscription termination notification and renew the subscription by sending a Subscription message by including all required elements and using the `all` attributes of the `EndDeviceList` as the `limit` parameter.
5. **[S]** Refuse the Subscription by responding back with an HTTP `400` response code
6. **[C]** By receiving the HTTP `400` response code for the requested subscription, the Client shall fallback to using polling for the `EndDeviceList` resource. Perform an HTTP GET request on the `EndDeviceList` resource periodically to check for changes.

Pass/Fail Criteria

- **[C, S]** Server successfully issued a Terminate notification message with correct status value as specified in the test setup.
- **[C, S]** Client successfully received the Terminate Notification message then renewed the subscription for the same resource by sending a Subscription request using the information specified in the test setup.
- **[C, S]** Server correctly refused the renewal subscription request and sent an HTTP `400` response code.
- **[C, S]** Client successfully received the HTTP `400` response code for its renewal Subscription request and fell back to periodic polling of the same resource using the polling intervals specified in the CSIP guide or specified by the Utility Interconnection Handbook.

Requirements Matrices

CSIP Requirements Matrix

Req. ID	Description	Test
G1.	Each DER Client SHALL connect to the utility in one and only one scenario.	(optional) No test; utility handbook policy
G2.	Although outside the scope of CSIP, security SHOULD be used in all non-IEEE 2030.5 interactions between the Aggregators, site hosts, GFEMS, and DERs and other entities receiving or transmitting DER related communications	(optional) No test; utility handbook policy
G3.	For DER Clients that have an IEEE 2030.5 certificate, the GUID SHALL be derived from this certificate (see section 5.2.1.2).	All tests
G4.	Implementers SHALL refer to each utility's Interconnection Handbook for requirements related to the creation, use or management of this identifier.	(optional) No test; utility handbook policy
G5.	Aggregators and DER Clients SHALL support IEEE 2030.5 based grouping and full lifecycle management of group relationships as defined within Section 5.2.3 and within each utility's Interconnection Handbook or program/contract requirements.	BASIC, UTIL, AGG, MAINT
G6.	Autonomous functions' default settings SHALL be changeable via IEEE 2030.5 DefaultDERControl communications.	BASIC, UTIL, AGG, MAINT
G7.	Modifications to default settings SHALL occur immediately upon receipt and have an indefinite duration.	BASIC, UTIL, AGG, MAINT
G8.	Scheduling Autonomous and Advanced Power Values and Modes SHALL be controllable via IEEE 2030.5 DERControl events	BASIC, UTIL, AGG, MAINT
G9.	Aggregators and DER Clients SHALL be responsible for assuring that all operations received from the utility are processed in the appropriate time sequence as specified by the utility.	BASIC, AGG
G10.	An Aggregator acting for its DERs and DER Clients SHALL be able to store at least 24 scheduled DER control events for each DER.	BASIC
G11.	In the absence of scheduled controls, DERs SHALL maintain a default control setting specified by interconnection tariffs or the utility Interconnection Handbook.	BASIC, UTIL, AGG, MAINT
G12.	Should there be a loss of communications, DERs SHALL complete any scheduled event and then revert to default settings or as determined by the site host or tariffs/contracts.	ERR
G13.	When commanded in a manner where two or more operations are possibly in conflict, the interpreting system SHALL operate against the control operation which has the highest priority subject to the systems capability, contracts and self-protection requirements.	(optional) No test; utility handbook policy
G14.	If avoidance of conflicting commands is not possible, the more recently received command SHOULD have precedence over the older command.	BASIC, UTIL, AGG, MAINT
G15.	In either case, it SHALL be the responsibility of the aggregator or DER Client to decide how to handle these two simultaneous controls.	BASIC, AGG
G16.	For Aggregators communications, notifications and call backs (subscription/notification) SHALL be used to limit system polling to the greatest extent practical.	Subscription-based
G17.	To simplify communication requirements for Direct DER Communications scenarios, unless specified otherwise in utility Interconnection Handbooks or programs/contracts, all communications SHALL be initiated by the DER Client (i.e., client-side initiation).	All tests
G18.	In Direct DER communication scenarios, the DER Client SHALL initiate communications with the utility according to a pre-defined polling and posting interval to ensure the DER has up to date settings and the utility understands the operational state of the DER.	BASIC, CORE
G19.	Unless specified in each utility's Interconnection Handbook, default polling and posting rates SHALL be as follows: • Polling of DERControls and DefaultDERControls (Direct DER Communication)– every 10 minutes • Posting monitoring information (Direct and Aggregator Mediated Communications)– every 5 minutes	CORE: polling
G20.	For DERs with an external SMCU, the SMCU SHALL transfer the DER control to the generating facility within 10 minutes of receiving the control from the server.	Out-of-scope

Req. ID	Description	Test
G21.	For DERs with a GFEMS, the GFEMS SHALL transfer the DER control to the DERs within 10 minutes of receiving the control from the server.	Out-of-scope
G22.	For DERs mediated by Aggregators, the Aggregator SHALL transfer the DER control to the DERs within 15 minutes of receiving the control from the server.	Out-of-scope
G23.	Aggregators acting for its DERs and DER Clients SHALL have the capability to report the monitoring data in Table 2.	CORE: mirrored metering
G24.	Aggregators acting for its DERs and DER Clients SHALL have the capability to include the data qualifiers in Table 3.	CORE: mirrored metering
G25.	All measurement SHALL include a date-time stamp.	CORE: mirrored metering
G26.	Unless otherwise specified in each utility's Interconnection Handbook or programs/contracts, Aggregators acting for its DERs and DER Clients SHALL report the monitoring data in Table 2 and MAY include the data qualifiers in Table 3	CORE: mirrored metering
G27.	For those situations where the DERs cannot provide Monitoring Data, the Aggregator acting for its DERs and DER Clients SHALL not send the data.	(optional) No test; utility handbook policy /PICS
G28.	Aggregators acting for its DERs and DER Clients SHALL have the capability to report the Nameplate Ratings and Adjusted Settings information shown in Table 4.	CORE: DER settings
G29.	Nameplate Ratings and Adjusted Settings SHOULD be reported once at start-up and whenever there is a change in value.	CORE: DER settings
G30.	Aggregators acting for its DERs and DER Clients SHALL have the capability to report the dynamic Operational Status Information shown in Table 5.	CORE: DER settings
G31.	Aggregators acting for its DERs and DER Clients SHALL have the capability to report the alarm data shown in Table 6 as they occur.	BASIC: alarms
G32.	All alarms and their "return to normal" messages SHALL include a date-time stamp along with the alarm type.	BASIC: alarms
P1.	The specific version of the protocol implemented SHALL be IEEE 2030.5-2018.	All tests
P2.	Utility servers, Aggregators, and DER Clients SHALL support all CSIP required IEEE 2030.5 function sets and resources in Table 7.	BASIC, CORE
P3.	Unless otherwise specified in the utility's Implementation Handbook, coordination of this time and rates for updating this time SHALL conform to the requirements of IEEE 2030.5-2018.	CORE: time
P4.	Aggregators acting for its DERs and DER Clients SHALL support the EndDevice:DER resources in Table 8 if the utility server makes them available.	CORE: EndDevice
P5.	Aggregators and DER Clients SHALL meet all IEEE 2030.5 mandatory requirements that are described in the standard for each of these sections/functions unless otherwise specified in utility Interconnection Handbooks or programs/contracts.	(optional) No test; utility handbook policy
P6.	HTTPS SHALL be used in all Direct and Aggregated communications scenarios.	CORE: security
P7.	Aggregators and DER Clients SHALL support the required IEEE 2030.5 security framework and other security frameworks as required by the utility Interconnection Handbook or programs/contracts.	CORE: security
P8.	TLS version 1.2 SHALL be used for all HTTPS transactions.	CORE: security
P9.	DER Clients SHALL support the IEEE 2030.5 cipher suite.	CORE: security
P10.	Aggregators SHALL also support the TLS_RSA_WITH_AES_256_CBC_SHA256 cipher suite or other cipher suites as specified by the utility Interconnection Handbook or programs/contracts.	(optional) No test; utility handbook policy
P11.	All utility servers, Aggregators, and DER Clients SHALL have a valid certificate.	CORE: security
P12.	A valid certificate SHALL be used in all IEEE 2030.5 TLS transactions.	CORE: security
P13.	Certificates for Aggregators and DER Clients SHALL only be provisioned upon completion of Conformance Testing.	No test; policy requirement
P14.	The GUID for both Aggregators and DERs SHALL be the IEEE 2030.5 Long Form Device Identifier (LFDI) which is based on the 20-byte SHA-256 hash of the device's certificate.	CORE: EndDevice
P15.	The certificates specified by each utility SHALL be used for authentication.	CORE: security
P16.	If authentication fails, the authenticator SHOULD issue a TLS Alert – Bad Certificate and close the connection.	CORE: security
P17.	For Aggregators and DER Clients, the authorization list SHALL be based on the LFDI since the SFDI may not provide enough collision protection for a large population (e.g. 1 million) of devices.	CORE: security

Req. ID	Description	Test
P18.	If the device is not on the authorization list, the utility server SHOULD return an HTTP error code (e.g. 404 – Not Found) to terminate the transaction.	(optional) No test; SHOULD requirement
P19.	The utility SHALL establish the permissions for read, write, control, and other interactions, based on agreements on which interactions are authorized between each DER and the utility.	(optional) No test; utility handbook policy
P20.	When an Aggregator accesses the EndDeviceList, the utility server SHALL only present EndDevices that are under the management of that Aggregator.	CORE: EndDevice
P21.	In the Direct DER Communications scenario, the GUID used to identify the DER Client SHALL be the DER's LFDI.	CORE: EndDevice
P22.	Implementers SHOULD refer to each utility's Interconnection Handbook or programs/contracts for more information needed to establish the LFDI.	(optional) No test; SHOULD requirement
P23.	Aggregators acting for its DERs and DER Clients SHALL track the DERProgram associated with that group.	AGG: event
P24.	Aggregators acting for its DERs and DER Clients SHALL support up to 15 DERPrograms simultaneously for each DER.	AGG: event
P25.	Aggregators acting for its DERs and DER Clients SHALL traverse all these links and lists to discover all DERPrograms the DER is required to track.	AGG: event
P26.	For each DER EndDevice, the utility server SHALL use one FSA to point to a DERProgramList containing all topology-based DERPrograms and MAY use additional FSAs to point to a DERProgramList containing non-topology-based DERPrograms.	AGG: event, all DER
P27.	DER Clients SHALL be capable of supporting 15 FSAs.	CORE: function set assignments
P28.	For the CSIP Direct Communication scenario, the DER Client SHALL only receive function set assignments for a single energy connection point reflecting the aggregate capabilities of the plant at its point of common coupling with the utility.	CORE: function set assignments
P29.	DER Clients SHALL use the IEEE 2030.5 mappings for the Grid DER Support Functions shown in Table 9.	BASIC: inverter control
P30.	DERControls are IEEE 2030.5 events and SHALL conform to all the event rules in Section 12.1.3 of IEEE 2030.5-2018.	AGG: event, all DER
P31.	Aggregators SHALL subscribe to each DERProgramList assigned to its DERs to discover changes in DERProgram:primacy.	AGG: subscription
P32.	Aggregators SHALL subscribe to the DERControlList of each DERProgram assigned to its DERs to discover new controls or changes to existing controls.	AGG: subscription
P33.	Aggregators SHALL subscribe to the DefaultDERControl of each DERProgram assigned to its DERs to discover changes to the default controls.	AGG: subscription
P34.	Unless otherwise specified in utility Interconnection Handbooks or programs/contracts to allow subscriptions, DER Clients SHALL poll to each DERProgram assigned to it to discover changes in DERProgram:primacy.	CORE: polling
P35.	Unless otherwise specified in utility Interconnection Handbooks or programs/contracts to allow subscriptions, DER Clients SHALL poll to the DERControlList of each DERProgram assigned to it to discover new controls or changes to existing controls.	CORE: polling
P36.	Unless otherwise specified in utility Interconnection Handbooks or programs/contracts to allow subscriptions, DER Clients SHALL poll to the DefaultDERControl of each DERProgram assigned to it to discover changes to the default controls.	CORE: polling
P37.	The utility MAY optionally specify a recommended polling rate for these resources using the DERProgramList:pollRate resource.	CORE: polling
P38.	If the polling rate is specified, DER Clients SHOULD poll at this rate.	CORE: polling
P39.	Aggregators SHALL subscribe to the following lists: • EndDeviceList • FunctionSetAssignmentsList of each of the DERs under its management • DERProgramList of each of the DERs under its management • DERControlList of each of the DERs under its management • DefaultDERControls of each of the DERs under its management	AGG: subscription
P40.	Aggregators MAY subscribe to other lists and instances, such as EndDevice, DERProgram, DERControl instances and others	
P41.	Aggregators acting for its DERs and DER Clients SHALL use the IEEE 2030.5 Metering Mirror function set to report metrology data.	CORE: mirrored metering

Req. ID	Description	Test
P42.	Aggregators acting for its DERs and DER Clients SHOULD post readings based on the MirrorUsagePoint:postRate resource.	
P43.	Aggregators acting for its DERs and DER Clients SHALL be able to report the information shown in Table 12.	CORE: mirrored metering
P44.	Aggregators acting for its DERs and DER Client SHALL be able to report the dynamic status information shown in Table 13.	CORE: mirrored metering
P45.	DER Clients SHALL be able to report alarm data shown in 14.	BASIC: alarms
P46.	The Aggregator SHOULD subscribe to the EndDeviceList to receive notifications for any additions or changes to the list.	AGG
P47.	The Aggregator SHOULD subscribe to each EndDevice instance under its control to receive notifications for any deletions of that instance.	AGG
P48.	For every inverter under its control, the Aggregator SHOULD subscribe to the list pointed to by EndDevice: FunctionSetAssignmentsListLink to receive notifications for any changes in the inverter's group assignments.	AGG
P49.	For every inverter under its control, the Aggregator SHOULD subscribe to all of the DERControlLists associated with its FSA groups and DERProgram assignments to receive notifications for any new or changed DERControl events.	AGG
P50.	For every inverter under its control, the Aggregator SHOULD subscribe to all of the DERPrograms associated with its FSA groups to receive notifications for changes to the DERProgram meta-data.	AGG
P51.	For every inverter under its control, the Aggregator SHOULD subscribe to all of the DERProgramLists associated with its FSA groups to receive notifications for additions, deletions, or changes to the list.	AGG
P52.	Maintenance of subscriptions is described previously for the IEEE 2030.5 Specification. In particular: • The Aggregator Client SHOULD renew its subscriptions periodically (e.g. every 24 hours) with the Utility Server. • The Aggregator Client SHOULD fall back to polling on perceived communications errors.	AGG

IEEE 2030.5 Requirements Matrix

This section contains a list of mandatory functionality specified in the IEEE 2030.5 specification grouped as described in the Testing Overview section of this specification.

The **Req. ID** column contains the assigned ID based on the grouping created in this specification. The **Section** column identifies the location of the requirement in the IEEE 2030.5 specification. The **Description** column provides a related excerpt from the IEEE 2030.5 specification, describing the specific requirement. The **Test IDs** column is a cross reference of the primary tests in this specification that test the requirement.

Req. ID	Section	Description	Test IDs
BASE.001	8.2.3	A resource serving device (i.e., all servers) SHALL implement the DeviceCapability resource.	CORE-007
BASE.002	8.4.2	However, clients SHALL NOT rely on this filtering behavior, as it is not mandatory server behavior; therefore, clients SHALL support the normal paging mechanism for List-type resources.	CORE-004
BASE.003	8.4.2	If no thresholds are specified by the server, then clients SHALL NOT POST updates to their LoadShedAvailability for the provider of that program.	
BASE.004	8.5.3	Clients that act on events SHALL determine if they are assigned into Function Set Assignments.	COMM-006, CORE-008, CORE-010
BASE.005	8.5.3	If a server supports Function Set Assignments it SHALL support a minimum of 1 Function Set Assignments for each HAN device registered to the server.	COMM-006, CORE-008, CORE-010
BASE.006	8.5.3	Clients SHALL support at least 6 function set instances (e.g., two DemandResponseProgram instances, one Time instance, one TariffProfile instance, and two MessagingProgram instances) assigned through 1 or more Function Set Assignments.	COMM-006, CORE-008, CORE-010
BASE.007	8.5.3	If a FunctionSetAssignments instance contains references to time-responsive function sets, it MUST also include a reference to a Time resource.	COMM-006, CORE-005, CORE-006, CORE-010
BASE.008	8.5.3	Clients SHALL identify which Function Set Assignments apply to them by querying the FunctionSetAssignments resource within their End Device function set instance (see the End Device section for more details).	COMM-006, CORE-008, CORE-010
BASE.009	8.5.3	Clients SHALL periodically poll their group assignments under their EndDevice resource (e.g., /edev/{#}/fsa), and the corresponding Function Set Assignments resource (e.g. /fsa/{#}), or SHALL subscribe to them to monitor for changes.	BASIC-002, COMM-006, CORE-003, CORE-008, CORE-010
BASE.011	8.5.3	Client devices that do not subscribe SHALL query at least once every 24 hours but SHALL NOT query more than once per hour.	
BASE.012	8.6.2	Notification resources typically are not exposed such that they can be read over the Smart Energy network, but devices that do choose to expose the resource(s) MUST obey the list ordering rules below.	
BASE.013	8.6.3.2	The resource representation returned in a notification SHALL be identical to that which would be returned via a GET request to the resource, subject to the limit parameter discussed below and per the rules stated earlier in this specification for representing resources.	CORE-018, CORE-019
BASE.014	8.6.3.2	Notification servers (subscription clients) SHALL support TLS as a server if they wish to receive secure notifications.	CORE-018, CORE-019
BASE.015	8.6.3.3	If both an upper threshold and a lower threshold are specified, the upper threshold SHALL be greater than the lower threshold, otherwise an error representation SHALL be returned.	
BASE.016	8.6.3.3	If neither a lower threshold nor an upper threshold is specified, then a server SHALL send a notification whenever the resource to which the client is subscribed changes.	
BASE.017	8.6.3.3	If a lower threshold and / or an upper threshold are specified, then a server SHALL send a notification whenever the appropriate value crosses (in either direction) the appropriate threshold for the resource to which the client is subscribed.	
BASE.018	8.6.3.4	Subscriptions SHALL be maintained on servers through power loss.	
BASE.019	8.6.3.4	Servers SHALL use the subscription parameters from the latest subscription renewal.	
BASE.020	8.6.3.4	If the URI of a resource changes, then subscriptions to that resource SHALL be terminated.	

Req. ID	Section	Description	Test IDs
BASE.021	8.6.3.4	For subscriptions to non-list resources, a notification SHALL be sent whenever the representation of the non-list resource changes.	CORE-018, CORE-019
BASE.022	8.6.3.4	For subscriptions to list resources, a notification SHALL be sent whenever any subordinate resources are added to or removed from the list, or if the representation of those subordinate resources change.	CORE-018, CORE-019
BASE.023	8.6.3.4	Servers implementing the subscription resource SHALL be capable of maintaining a minimum of 1 subscription and servers implementing the subscription resource SHOULD support at least 1 subscription per device per subscribable resource.	CORE-018, CORE-019
BASE.024	8.6.3.4	If a server implementing the subscription resource is unable to accept a subscription, the server SHALL return an error resource representation indicating the specific error (e.g., element does not support conditional subscription) with an HTTP response code of 400.	AGG-001, CORE-001, CORE-010, CORE-015, CORE-018, UTIL-001
BASE.025	8.6.3.4	When a server removes or terminates a subscription, it SHALL send the client a terminate subscription (except after an error from an undesired subscription, as mentioned in rule 14 below).	AGG-001, CORE-001, CORE-010, CORE-018, UTIL-001
BASE.026	8.6.3.4	The server SHALL send a Notification to the client's ""Post URI"" for the affected subscription.	AGG-001, CORE-001, CORE-010, CORE-015, CORE-018, UTIL-001
BASE.027	8.6.3.4	The Resource contained in the Notification SHALL be a Subscription, containing a Status element identifying the reason for terminating the subscription.	AGG-001, CORE-001, CORE-010, CORE-018, UTIL-001, MAINT-006
BASE.028	8.6.3.4	If a client receives a notification for an undesired subscription, the client SHALL return an HTTP 400 error.	AGG-001, CORE-001, CORE-010, CORE-015, CORE-018, UTIL-001, MAINT-006
BASE.029	8.6.3.4	Upon receipt of such an error, the server SHALL remove the subscription without notification.	AGG-001, CORE-001, CORE-010, CORE-015, CORE-018, UTIL-001, MAINT-006
BASE.030	*8.7.2*	*Clients of any function set that can utilize the Response function set (e.g., DRLC, DER, Price, Messaging, Flow Reservation) SHALL implement a Response function set client.[SR(EC46].*	*CORE-022*
BASE.031	8.7.2	If a response is desired to an event, then the event SHALL provide, in the replyTo field, a URI indicating the location of where the responses are to be posted.	*CORE-022*
BASE.032	8.7.2	The client SHALL POST the responses to the indicated URI based on the rules indicated by the responseRequired bitfield of the event.	*CORE-022*
BASE.033	8.7.2	If the server supports the GET method for the response function set it SHALL minimally support 1 response for each function set for which it accepts responses.	*CORE-022*
BASE.034	B.1.2.1	This attribute SHALL be present if the href is a local or relative URI.	CORE-022
BASE.035	B.1.2.1	Responses shall be posted to the collection specified in ""replyTo"".	CORE-022
BASE.036	B.1.2.1	If the resource has a deviceCategory field, devices that match one or more of the device types indicated in deviceCategory SHALL respond according to the rules listed below.	
BASE.037	B.1.2.1	If the category does not match, the device SHALL NOT respond.	
BASE.038	B.1.2.1	If the resource does not have a deviceCategory field, a device receiving the resource SHALL respond according to the rules listed below.	
BASE.039	B.1.2.1	0 - End device shall indicate that message was received.	
BASE.040	B.1.2.1	1 - End device shall indicate specific response.	
BASE.042	B.1.2.3	DstRuleType: Bit map encoded rule from which is calculated the start or end time, within the current year, to which daylight savings time offset must be applied.	
BASE.043	B.1.2.3	A master resource identifier. The IANA PEN [PEN] provider ID SHALL be specified in bits 0-31, the least-significant bits, and objects created by that provider SHALL be assigned unique IDs 5361 with the remaining 96 bits.	
BASE.044	B.1.2.3	mRIDType: Except for this special reserved identifier, each modification of an object (resource) representation MUST have a different ""version"".	
BASE.045	B.1.2.3	PowerOfTenMultiplierType Object (Int8) Any integer between -9 and 9 SHALL be supported, indicating the power of ten multiplier for the units.	BASIC-002, CORE-010

Req. ID	Section	Description	Test IDs
BASE.046	B.1.2.3	RoleFlagsType Object (HexBinary16)Bit 0 - isMirror - SHALL be set if the server is not the measurement device.	
BASE.047	B.1.2.3	RoleFlagsType Object (HexBinary16)Bit 1 - isPremisesAggregationPoint - SHALL be set if the UsagePoint is the point of delivery for a premises.	
BASE.048	B.1.2.3	RoleFlagsType Object (HexBinary16)Bit 2 - isPEV - SHALL be set if the usage applies to an electric vehicle.	
BASE.049	B.1.2.3	RoleFlagsType Object (HexBinary16)Bit 3 - isDER - SHALL be set if the usage applies to a distributed energy resource, capable of delivering power to the grid.	
BASE.050	B.1.2.3	RoleFlagsType Object (HexBinary16)Bit 4 - isRevenueQuality - SHALL be set if usage was measured by a device certified as revenue quality.	
BASE.051	B.1.2.3	RoleFlagsType Object (HexBinary16)Bit 5 - isDC - SHALL be set if the usage point measures direct current.	
BASE.052	B.1.2.3	RoleFlagsType Object (HexBinary16)Bit 6 - isSubmeter - SHALL be set if the usage point is not a premises aggregation point.	
BASE.053	B.1.2.3	Version SHALL indicate a distinct identifier for each revision of an IdentifiedObject.	
BASE.054	B.1.2.3	If not specified, a default version of ""0"" (initial version) SHALL be assumed.	
BASE.055	B.1.2.3	Upon modification of any IdentifiedObject, the mRID SHALL remain the same, but the version SHALL be incremented.	
BASE.056	B.1.2.4	In order to limit internal storage, implementations SHALL reduce the length of strings using multi-byte characters so that the string may be stored using ""maxLength"" octets in the given encoding.	
BASE.061	B.1.2.4	For all string types, in order to limit internal storage, implementations SHALL reduce the length of strings using multi-byte characters so that the string may be stored using ""maxLength"" octets in the given encoding.	
BASE.062	B.1.5	Indicates a condition that must be satisfied for the Notification to be triggered.	
BASE.063	B.1.5	subscribedResource attribute (anyURI) The resource for which the subscription applies. Query string parameters SHALL NOT be specified when subscribing to list resources.	AGG-001, CORE-018, CORE-019, ERR-002
BASE.064	B.1.5	Should a query string parameter be specified, servers SHALL ignore them.	AGG-001, CORE-018, CORE-019, ERR-002
BASE.065	B.1.5	For list resources, if a limit of '0' is specified, then notifications SHALL contain a list resource with results='0' (equivalent to a simple change notification).	AGG-001, CORE-018, CORE-019, ERR-002
BASE.066	B.1.5	For list resources, if a limit greater than '0' is specified, then notifications SHALL contain a list resource with results equal to the limit specified (or less, should the list contain fewer items than the limit specified or should the server be unable to provide the requested number of items for any reason) and follow the same rules for list resources (e.g., ordering).	AGG-001, CORE-018, CORE-019, ERR-002
BASE.067	B.1.5	For non-list resources, if a limit of '0' is specified, then notifications SHALL NOT contain a resource representation (equivalent to a simple change notification).	AGG-001, CORE-018, CORE-019, ERR-002
BASE.068	B.1.5	For non-list resources, if a limit greater than '0' is specified, then notifications SHALL contain the representation of the changed resource.	AGG-001, CORE-018, CORE-019, ERR-002
BASE.069	B.1.5	notificationURI attribute (anyURI) The resource to which to post the notifications about the requested subscribed resource. Because this URI will exist on a server other than the one being POSTed to, this attribute SHALL be a fully-qualified absolute URI, not a relative reference.	AGG-001, CORE-018, CORE-019, ERR-002
BASE.070	B.1.5	This attribute SHALL be a fully-qualified absolute URI, not a relative reference.	
BASE.072	B.1.6	When overriding within the allowed override duration, the client SHALL send a partial opt-out (Response status code 8) for partial opt-out upon completion, with the total time the event was overridden (this attribute) populated.	CORE-022
BASE.073	B.1.6	The client SHALL send a no participation status response (status type 10) if the user partially opts-out for longer than EndDeviceControl.overrideDuration.	
CFG.001	B.1.12	dstStartRule: Result of dstEndRule must be greater than result of dstStartRule.	
DER.001	10.9.4.1	DERProgram server devices SHALL be capable of internally storing and supporting at least one DERProgram instance.	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012, CORE-013, MAINT-005, UTIL-004
DER.002	10.9.4.1	Each DERProgram instance SHALL be uniquely identified by an mRID.	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012,

Req. ID	Section	Description	Test IDs
			CORE-013, MAINT-005, UTIL-004
DER.003	10.9.4.1	DER client devices SHALL be capable of internally storing and supporting at least one DERProgram instance.	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012, CORE-013, MAINT-005, UTIL-004
DER.004	10.9.4.2	At any point in time a DER client is managed by a single DERControl Event, which SHALL supersede any previous Event.	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012, CORE-013, MAINT-005, UTIL-004
DER.005	10.9.4.2	If there are no active Events, the DER client SHALL be managed by the DefaultDERControl instance exposed by the preferred DERProgram.	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012, CORE-013, MAINT-005, UTIL-004
DER.006	10.9.4.2	DERProgram server devices SHALL be capable of internally storing and supporting at least five unique DERControl instances, which MAY be distributed among multiple programs.	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012, CORE-013, MAINT-005, UTIL-004
DER.007	10.9.4.2	Each DERProgram SHALL internally store a single DefaultDERControl instance that defines the default behavior of associated DER clients when no active Events are available.	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012, CORE-013, MAINT-005, UTIL-004
DER.008	10.9.4.2	Each DERControl and DefaultDERControl instance SHALL be uniquely identified by an mRID.	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012, CORE-013, MAINT-005, UTIL-004
DER.009	10.9.4.2	DER client devices SHALL be capable of internally storing and supporting at least one DERControl instance and a single DefaultDERControl instance for each stored DERProgram.	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012, CORE-013, MAINT-005, UTIL-004
DER.010	10.9.4.3	Each DERCurve SHALL contain a defined curveType value that associates the curve with a given control mode and implicitly defines the units of measure that apply to its curveData points.	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012, CORE-013, MAINT-005, UTIL-004
DER.011	10.9.4.3	A DERCurve SHALL specify an array of one or more curveData points.	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012, CORE-013, MAINT-005, UTIL-004
DER.012	10.9.4.3	Implementations SHALL support a minimum of four curves having a minimum of 10 points per curve for each curve-based DERControl mode supported, unless otherwise specified.	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012, CORE-013, MAINT-005, UTIL-004
DER.013	10.9.4.3	The currently executing DERProgram SHALL be referenced by the DER's CurrentDERProgramLink.	
DER.014	10.9.4.4.2	A modified rating SHALL have a corresponding setting.	
DER.015	10.9.5	For example, if a generator type DER can deliver reactive power and supports fixed VAr control mode then it must include rtgVAr and setMaxVAr.	
DER.016	10.9.5	Each unique DER instance SHALL link to a DERCapability instance that SHALL contain type, modesSupported, rtgA, and rtgW attributes. T.	CORE-014, CORE-016
DER.017	10.9.5	However, if a storage DER also supports discharge mode then it MUST use rtgMaxChargeRate to expose its charging rate and MAY use either rtgMaxDischargeRate (if the DER supports simultaneous generation and discharge) or rtgW to expose its maximum discharging rate.	CORE-014, CORE-016
DER.018	B.1.21	When present, this value SHALL update the value of the corresponding setting (DERSettings::setESDelay).	

Req. ID	Section	Description	Test IDs
DER.019	B.1.21	When present, this value SHALL update the value of the corresponding setting (DERSettings::setESHighFreq).	
DER.020	B.1.21	When present, this value SHALL update the value of the corresponding setting (DERSettings::setESHighVolt).	
DER.021	B.1.21	When present, this value SHALL update the value of the corresponding setting (DERSettings::setESLowFreq).	
DER.022	B.1.21	When present, this value SHALL update the value of the corresponding setting (DERSettings::setESLowVolt).	
DER.023	B.1.21	When present, this value SHALL update the value of the corresponding setting (DERSettings::setESRandomDelay).	
DER.024	B.1.21	When present, this value SHALL update the value of the corresponding setting (DERSettings::setGradW).	
DER.025	B.1.21	When present, this value SHALL update the value of the corresponding setting (DERSettings::setSoftGradW).	
DER.026	B.1.21	setMaxVAr: Set limit for maximum reactive power delivered by the DER (in var). SHALL be a positive value <= rtgVAr (default).	CORE-014, CORE-016
DER.027	B.1.21	setMaxVArNeg: If present, SHALL be a negative value >= rtgVArNeg (default).	CORE-014, CORE-016
DER.028	B.1.21	setMinPF: SHALL be >= rtgMinPF (default).	CORE-014, CORE-016
DER.029	B.1.21	setMinPFNeg:If present, SHALL be <= rtgMinPFNeg (default).	CORE-014, CORE-016
DER.030	B.1.21	Minimum Power Factor displacement capability of the DER; SHALL be a positive value between 0.0 (typically > 0.7) and 1.0.	CORE-014, CORE-016
DER.031	B.1.21	Minimum Power Factor displacement capability of the DER; SHALL be a negative value between 0.0 (typically < -0.7) and -0.9999.	CORE-014, CORE-016
DER.032	B.1.21	If present, rtgOverExcitedPF SHALL be present.	
DER.033	B.1.21	If present, rtgUnderExcitedPF SHALL be present.	
DER.034	B.1.21	opModFixedPFInjectW: The PF sign (which SHALL be interpreted according to the EEI convention, where unity PF is considered unsigned) indicates the direction of reactive power flow.	CORE-014, CORE-016
DER.035	B.1.21	opModFixedPFInjectW: The actual displacement SHALL be within the limits established by setMinPF and setMinPFNeg.	
DER.036	B.1.21	functionSet attribute (UInt8) If the profileID indicates this is the Smart Energy Profile, the functionSet is defined by the Zigbee Alliance and SHALL be one of the values from the table below (IEEE 2030.5 function set identifiers).	
DER.037	B.1.21	The meaning of the y value is determined by yRefType and must be one of %setMaxW, %setMaxVar, or %statVarAvail.	
DER.038	B.1.21	opModWattPF: The PF output setting is a signed displacement in y value (PF sign SHALL be interpreted according to the EEI convention, where unity PF is considered unsigned).	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012, CORE-013, MAINT-005, UTIL-004
DER.040	B.1.21	DERControlType: Bit positions SHALL be defined as follows:	
DER.042	B.1.21	displacement: Sign SHALL be interpreted according to the EEI convention.	AGG-002, BASIC-002, BASIC-003, BASIC-016, CORE-012, CORE-013, MAINT-005, UTIL-004
DI.001	9.2.3	Any device that implements an HTTP server to serve resources SHALL implement the DeviceInformation resource (this does not apply to devices that only serve Notification / NotificationList resources for the purpose of receiving subscription notifications).	
DNS.001	7.1	A server SHALL assign a unique label of up to 63 bytes in UTF-8 format for each DNS SRV / TXT record pair that it advertises.	COMM-001
DNS.002	7.1	Should a name conflict occur, a device SHALL assign itself a new name until conflicts are resolved.	COMM-001
DNS.003	7.1	When an SFDI is used as part of a DNS-SD label, it SHALL be represented as 12 decimal digits including leading zeros (if any) as well as the checksum digit and SHALL NOT include embedded hyphens.	COMM-001
DNS.004	7.2	The Service Name used with IEEE 2030.5 DNS-SD SHALL be smartenergy and has been registered appropriately with IANA [IANA SN].	COMM-001

Req. ID	Section	Description	Test IDs
DNS.005	7.3	IEEE 2030.5 SHALL use a single TXT record format.	COMM-001
DNS.006	7.3	txtvers SHALL be the first key in the TXT record.	COMM-001
DNS.007	7.3	For version 2.0 of the Smart Energy Profile, the value of the txtvers key SHALL be 1.	COMM-001
DNS.008	7.3	If it is found in a response to be other than 1, the TXT record SHALL be ignored.	COMM-001
DNS.009	7.3	The txtvers key SHALL be present with a non-empty value.	COMM-001
DNS.010	7.3	Clients SHALL silently discard TXT records with txtvers keys that are not present with a non-empty value of 1.	COMM-001
DNS.011	7.3	Unknown key=value pairs in a response SHALL be ignored.	COMM-001
DNS.012	7.3	The dcap key SHALL provide the relative reference used to locate the device's DeviceCapability resource and SHALL include the leading slash of the given path.	COMM-001
DNS.013	7.3	The dcap key SHALL be present with a non-empty value.	COMM-001
DNS.014	7.3	Clients SHALL silently discard TXT records with dcap keys that are not present with a non-empty value representing the URI for the server DeviceCapability resource.	COMM-001
DNS.015	7.3	The path key is used in responses to subtype queries (see below) and SHALL provide the relative reference used to locate the base path of a specified function set or resource, including the leading slash '/' of the given path.	COMM-001
DNS.016	7.3	The path key SHALL be omitted in response to service name queries.	COMM-001
DNS.017	7.3	The path key SHALL be present with a non-empty value representing the URI satisfying the subtype query.	COMM-001
DNS.018	7.3	Clients SHALL ignore path keys included with service name query responses where the path key is supplied with no value or present with an empty value.	COMM-001
DNS.019	7.3	Servers supporting HTTPS using a non-default port SHALL indicate the port number by including the https key with a non-empty value representing the supported HTTPS port number.	COMM-001
DNS.020	7.3	Clients SHALL use HTTP for the service if the https key is not present.	COMM-001
DNS.021	7.3	Clients SHALL use HTTPS using the default port for the service if the https key is present with no value or present with an empty value.	COMM-001
DNS.022	7.3	Clients SHALL use HTTPS using the port number indicated if the https key is present with a non-empty value.	COMM-001
DNS.023	7.3	The default value SHALL be the supported Extensibility Level of the server.	COMM-001
DNS.024	7.3	If ""+S[i]"" is provided in the level key, the server SHALL support either of ""-S[i]"" or ""+S[i]"" as negotiated through the HTTP Accept header.	COMM-001
DNS.025	7.3	The level key SHALL be present with a non-empty value.	COMM-001
DNS.026	7.3	Clients SHALL silently discard TXT records with level keys that are not present with a non-empty value representing the server's schema level extensibility indicator.	COMM-001
DNS.027	7.4	A server SHALL register a PTR record with a subtype name as defined below for each function set that it advertises for discovery.	COMM-001
DNS.028	7.4	In addition, the server SHALL register a unique SRV / TXT record pair with an <Instance> as defined in Section 7.1 for each function set that is referenced by the subtype PTR record.	COMM-001
DNS.029	7.4	All SRV records registered on a given device SHALL contain identical values.	COMM-001
DNS.030	7.4	The port value contained in the SRV record SHALL be specified for the default (http) scheme.	COMM-001
DNS.031	7.4	If a secure connection is required for the function set or resource, then the https key SHALL be present in the TXT record as specified in Section 7.3.	COMM-001
DNS.032	7.4	The server SHALL register exactly one PTR record with the name ""_smartenergy._tcp.site."".	COMM-001
DNS.033	7.4	The TXT record of the SRV / TXT record pair referenced by a subtype PTR record SHALL conform to the definition given in Section 7.3 and SHALL contain the relative reference for the base path of the function set that corresponds to the specified subtyp.	COMM-001
DNS.034	7.4	Subtype strings SHALL NOT begin with an underscore (see Table 17).	COMM-001
DNS.035	7.4	For example, the subtype name for the meter usage point function set shall be composed as ""upt._sub._smartenergy._tcp.site."".	COMM-001

Req. ID	Section	Description	Test IDs
DNS.036	7.4	Other subtype names SHALL (except as noted below) be composed per the meter usage point example.	COMM-001
DNS.037	7.4	In this case, the subtype name SHALL consist of the string edev, concatenated with a hyphen and the remote device's SFDI, (see Section 8.3.2) and followed by ""_sub._smartenergy._tcp.site"".	COMM-001
DNS.038	7.4	The first part SHALL consist of an identifier unique across End Device instances on this server.	COMM-001
DNS.039	7.4	The second part SHALL be the edev subtype string.	COMM-001
DNS.040	7.5	It should be noted that clients SHALL dynamically discover the URI(s) of the resource(s) of interest, as the URI(s) MAY vary from server to server and MAY occasionally vary over the lifetime of a given server.	COMM-001
DNS.041	7.5	Clients SHALL rediscover URIs upon notification of the server DNS-SD record change or a request fails with a 404 (Not Found) error.	COMM-001
DNS.042	7.5	Clients SHALL in the case of redirects, follow the guidance in the HTTP specification [RFC 2616].	COMM-001
EVENT.001	10.1.3.2	Clients that act on events that do not subscribe to their Event lists SHALL poll the lists for new Events at least once every 15 minutes and SHOULD poll at least every 5 minutes.	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.002	10.1.3.2	Clients SHALL monitor the active Event(s) for status changes at least once every 15 minutes and SHOULD monitor at least once every 5 minutes.	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.003	10.1.3.2	The duration of the EndDeviceControl is provided in seconds using the duration attribute. Events SHALL NOT be longer than 86,400 seconds (1 day).	
EVENT.004	10.1.3.2	Editing Events SHALL NOT be allowed except for updating status. Service providers SHALL cancel Events that they wish clients to not act upon and / or provide new superseding Events.	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.005	10.1.3.2	For function sets with direct control and the Pricing function set, clients SHALL NOT simultaneously execute or report execution of multiple simultaneous Events. (e.g., Nested Events and Overlapping Events).	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.006	10.1.3.2	A client SHALL consider the current Event complete if a superseding Event is started.	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.007	10.1.3.2	When comparing two Nested Events or Overlapping Events, from servers with the same primacy the creationTime element SHALL be used to determine which Event is newer and therefore supersedes the older.	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.008	10.1.3.2	When a Nested Event completes the containing / superseded Event SHALL NOT be reinstated and remain in a superseded state.	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.009	10.1.3.2	Clients SHALL verify the EventStatus of an Event before acting upon it.	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.010	10.1.3.2	If the EventStatus potentiallySupercededTime has changed since last checked, and if the EventStatus type is ""Partially Superseded"", clients SHALL check all Events from that function set instance that may supersede the original Event.	
EVENT.011	10.1.3.2	When a client receives an Event with the Specified End Time in the past (Specified End Time < Current Time), this Event SHALL be ignored.	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.012	10.1.3.2	For function sets with direct control, if the Event responseRequired indicates, clients SHALL POST a Response to the replyTo URI with a Status of ""Rejected - Event was received after it had expired"".	
EVENT.013	10.1.3.2	When a client receives an Event and calculates an Effective Start Time (Start Time + Start Randomization) in the past and a Specified End Time in the future ((Effective Start Time < Current Time) AND (Specified End Time > Current Time)), the client SHALL begin the Event using the current time and the absolute value of Start Randomization.	CORE-021
EVENT.014	10.1.3.2	For response reporting purposes, the start time SHALL be reported as the Current Time plus applied Start Randomization applied.	CORE-021
EVENT.015	10.1.3.2	For Event duration purposes, the Specified End Time SHALL be preserved, and any duration randomization attributes SHALL be applied to the abbreviated duration.	CORE-021
EVENT.016	10.1.3.2	Depending on the state of the Event (scheduled or active), one of the following steps SHALL take place:	

Req. ID	Section	Description	Test IDs
EVENT.017	10.1.3.2	If the previous Event is scheduled and not active and if the Event responseRequired indicates, the client SHALL POST a Response (referencing the previous Event) with the Status of ""The Event has been superseded"".	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.018	10.1.3.2	If the previous Event is active, the client SHALL change directly from its current state to the requested state at the effective start time of the Overlapping Event.	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.019	10.1.3.2	If the Event responseRequired indicates, the client SHALL POST a response (referencing the previous Event) with a Status of 'The event has been superseded' at the effective start time of the Overlapping Event.	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.020	10.1.3.2	Randomization SHALL NOT cause Event conflicts or unmanaged gaps.	
EVENT.021	10.1.3.2	For Successive Events clients SHALL use the earlier Event 's Effective End Time as the Effective Start Time of the later Event.	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.022	10.1.3.2	Randomization SHALL NOT artificially create a gap between Successive Events.	CORE-021
EVENT.023	10.1.3.2	a) Those clients whose deviceCategory is not listed in the future Event but whose deviceCategory was included in the original Event SHALL continue to execute per the parameters of the original Event.	
EVENT.024	10.1.3.2	Rule #3 continues to apply. Servers SHALL NOT edit the original Event but SHALL maintain all Events in their entirety.	
EVENT.025	10.1.3.2	c) A server SHALL set the potentiallySuperseded flag when the Event is superseded for any of the device categories and update the potentiallySupersededTime.	
EVENT.026	10.1.3.2	When an Event is removed from the server (e.g., due to limited storage space for the Event list) clients SHALL NOT assume the Event has been cancelled.	BASIC-015
EVENT.027	10.1.3.2	Client devices SHALL only act on a cancellation as indicated in the rules above or an update to the Event's Status attribute.	BASIC-015
EVENT.028	10.1.3.2	a) Those clients whose control is not listed in the future Event but whose control was included in the original Event SHALL continue to execute per the parameters of the original Event.	BASIC-024, BASIC-025, BASIC-026, AGG-010, AGG-011, AGG-012
EVENT.030	10.1.3.2	c) A server SHALL set the potentiallySuperseded flag when the Event is superseded for any of the controls and update the potentiallySupersededTime.	
EVENT.031	B.1.2.2	Events SHALL be executed relative to the time of the server, as described in the Time function set section 9.1.	BASIC-015
EVENT.032	B.1.2.2	The server SHALL set the event to this status when the event is first scheduled and persist until the event has become active or has been cancelled.	BASIC-015
EVENT.033	B.1.2.2	For events with a start time less than or equal to the current time, this status SHALL never be indicated, the event SHALL start with a status of ""Active"".	BASIC-015
EVENT.034	B.1.2.2	1 = Active. This status indicates that the event is currently active. The server SHALL set the event to this status when the event reaches its earliest Effective Start Time.	BASIC-016 to BASIC-026, AGG-002 to AGG0-012
EVENT.035	B.1.2.2	When events are cancelled, the Status.dateTime attribute SHALL be set to the time the cancellation occurred, which cannot be in the future.	
EVENT.036	B.1.2.2	Client devices SHALL be aware of Cancelled events, determine if the Cancelled event applies to them, and cancel the event immediately if applicable.	
EVENT.037	B.1.2.2	Client devices SHALL be aware of Cancelled with Randomization events, determine if the Cancelled event applies to them, and cancel the event immediately, using the larger of (absolute value of randomizeStart) and (absolute value of randomizeDuration) as the end randomization, in seconds.	
EVENT.038	B.1.2.2	This Status.type SHALL NOT be used with TextMessage, since multiple text messages can be active.	
EVENT.039	B.1.2.2	Servers SHALL mark an event as Superseded at the earliest Effective Start Time of the overlapping event.	
EVENT.040	B.1.2.2	Client devices encountering a Superseded event SHALL terminate execution of the event immediately and commence execution of the new event immediately, unless the current time is within the start randomization window of the superseded event, in which case the client SHALL obey the start randomization of the new event.	
EVENT.042	B.1.2.2	The dateTime attribute will provide a timestamp of when the request status was set. dateTime MUST be set to the time at which the status change occurred, not a time in the future or past.	BASIC-016 to BASIC-026, AGG-002 to AGG0-012

Req. ID	Section	Description	Test IDs
EVENT.043	B.1.2.2	An Event that can indicate time ranges over which the start time and duration SHALL be randomized.	
EVENT.044	B.1.2.2	randomizeDuration: Number of seconds boundary inside which a random value must be selected to be applied to the associated interval duration, to avoid sudden synchronized demand changes.	CORE-021
EVENT.045	B.1.2.2	randomizeStart: Number of seconds boundary inside which a random value must be selected to be applied to the associated interval start time, to avoid sudden synchronized demand changes.	CORE-021
FILE.001	9.7.1.3	An example use of the activation state is the scenario in which an operator first must load and confirm load of new software images to a large set of devices (as non-activated, not running) before setting that software image to an activated state (the running image).	
FILE.002	9.7.1.3	A FileList MUST contain File resources for each file made available by the FS for download by LDs.	
FILE.003	9.7.1.3	The LD MUST first determine if there exists a FunctionSetAssignment, containing a FileList, for the device.	
FILE.004	9.7.1.3	If such a FileList exists, the LD MUST use this FileList exclusively.	
FILE.005	9.7.1.3	If a Time resource is specified within this same FunctionSetAssignment, the LD MUST use this Time resource as the reference for file activation time.	
FILE.006	9.7.1.3	If the LD does not locate a FunctionSetAssignment directing the LD to a specific FileList, the LD MUST attempt discovery of a FileList provided by any device's DeviceCapabilities resource.	
FILE.007	9.7.1.3	An LD MUST poll for available files at least once every 24 hours.	
FILE.008	9.7.1.3	An LD SHALL provide the ability to fully receive and store a non-activated file while the current activated file remains operational.	
FILE.009	9.7.1.3	For example, an LD SHALL be able to load a new software image (a non-activated file) while the current software image executes.	
FILE.010	9.7.1.3	An FS MUST be able to process the Range entity header within HTTP GET requests for file content and MUST support Range entity header processing for at least a single range of bytes.	
FILE.011	9.7.1.3	An FS MUST include the Content-Range entity header within HTTP responses for file content when an LD requests a specific range.	
FILE.012	9.7.1.3	An FS MUST maintain a unique entity-tag value for each version of a file referenced by a specific URI.	
FILE.013	9.7.1.3	Per [RFC 2616], the entity-tag value MUST change if the underlying bits of the file change.	
FILE.014	9.7.1.3	The FS MUST include an Etag entity header (indicating the file entity-tag value) in all GET responses containing file content.	
FILE.015	9.7.1.3	An LD MUST detect a change in file Etag value.	
FILE.016	9.7.1.3	An LD MUST abort the file load if the Etag has changed and SHOULD restart the file load (now based on the new Etag).	
FILE.017	9.7.1.3	For previously loaded, non activated files for which an activation time has not been acquired, the LD MUST poll the File resource at least once every 24 hours for inclusion of a file activation time.	
FILE.018	9.7.1.3	If the LD is unable to acquire an activation time after 5 attempts, the LD MUST cease any further attempts and consider the activation failed.	
FILE.019	9.7.1.3	If the LD does not support processing of the Retry-After entity header, the LD MUST wait at least 30 seconds before retrying the file content request.	
FILE.020	9.7.1.3.2	Files SHALL be signed, and signatures verified using approved algorithms specified in [NIST SP800-131A], [NIST SP800-131B], and [NIST SP800-131C].	
FILE.021	9.7.1.3.2	Cryptographic strength MUST be at least the minimum specified in [ZB 09-5449] Section 9.3 (which is 128-bit).	
FILE.022	9.7.1.3.2	The signature of the file SHALL be calculated over the entire contents of the file excluding the signature octets.	
FILE.023	9.7.1.3.3	File loads containing IEEE 2030.5 security artifacts (type = 1 or any manufacturer defined file type containing credentials, key material, etc.) MUST be protected to at least the current level of security associated with the artifact being loaded.	
FILE.024	9.7.1.3.3	Such a file load MUST be secured using an HTTPS connection.	

Req. ID	Section	Description	Test IDs
FILE.025	9.7.1.3.4	All of the query parameter values MUST adhere to identical syntax and semantics as used for the correspondingly named elements of the File resource described in [ZB 13-0201].	
FILE.026	9.7.1.3.4	An FS MUST be able to process all file query parameters.	
FILE.027	9.7.1.3.4	File query parameters MUST be applied to a File List before the standard list query parameters (i.e., the standard query parameters are used to page the list resulting after the file query parameters are applied).	
FILE.028	9.7.1.3.4	The set of file query parameters MUST be interpreted as a Boolean expression, with each file query parameter considered a clause of that expression, and each clause connected by a logical AND operand.	
FILE.029	9.7.1.3.4	Each file query parameter MUST provide an exact match with its File equivalent for its clause to evaluate to true.	
FILE.030	9.7.1.3.4	There is one exception to this case: the mfVer MUST be evaluated as a GREATER THAN logical operand versus its File equivalent.	
FILE.031	9.7.1.3.4	Omission of a query parameter MUST be interpreted as a match, regardless the File's corresponding value.	
FILE.032	9.7.1.4	If an LD does implement the LogEvents for this function set, the LD SHALL implement all of the following LogEvents.	
FILE.033	B.1.13	This element MUST be set to the date/time at which this file is activated.	
FILE.034	B.1.13	If the activation time is less than or equal to current time, the LD MUST immediately place the file into the activated state (in the case of a firmware file, the file is now the running image).	
FILE.035	B.1.13	If the activation time is greater than the current time, the LD MUST wait until the specified activation time is reached, then MUST place the file into the activated state.	
FILE.036	B.1.13	Omission of this element means that the LD MUST NOT take any action to activate the file until a subsequent GET to this File resource provides an activateTime.	
FILE.037	B.1.13	fileURI: This element MUST be set to the URI location of the file binary artifact.	
FILE.038	B.1.13	Lfdi: This element MUST be set to the LFDI of the device for which this file in targeted.	
FILE.039	B.1.13	mfHwVer: This element MUST be set to the hardware version for which this file is targeted.	
FILE.040	B.1.13	mfID: This element MUST be set to the manufacturer's Private Enterprise Number (assigned by IANA).	
FILE.041	B.1.13	mfModel: This element MUST be set to the manufacturer model number for which this file is targeted.	
FILE.042	B.1.13	mfSerNum: This element MUST be set to the manufacturer serial number for which this file is targeted.	
FILE.043	B.1.13	mfVer: This element MUST be set to the software version information for this file.	
FILE.044	B.1.13	size: This element MUST be set to the total size (in bytes) of the file referenced by fileURI.	
FILE.045	B.1.13	type: A value indicating the type of the file. MUST be one of the following values:.	
FILE.046	B.1.13	activateTime: Omission of or presence and value of this element MUST exactly match omission or presence and value of the activateTime element from the File resource.	
FILE.047	B.1.13	loadPercent: This element MUST be set to the percentage of the file, indicated by FileLink, that was loaded during the latest load attempt.	
FILE.048	B.1.13	loadPercent: This value MUST be reset to 0 each time a load attempt is started for the File indicated by FileLink.	
FILE.049	B.1.13	loadPercent: This value MUST be increased when an LD receives HTTP response containing file content.	
FILE.050	B.1.13	loadPercent: This value MUST be set to 100 when the full content of the file has been received by the LD.	
FILE.051	B.1.13	nextRequestAttempt: This element MUST be set to the time at which the LD will issue its next GET request for file content from the File indicated by FileLink.	
FILE.052	B.1.13	requestFailCount: This value MUST be reset to 0 when FileLink is first pointed at a new File.	
FILE.053	B.1.13	request503Count: This value MUST be incremented each time an LD receives a 503 error from the FS.	

Req. ID	Section	Description	Test IDs
FILE.055	B.1.13	This value MUST be incremented each time a GET request for file content failed. 503 errors MUST be excluded from this counter.	
FILE.057	B.1.13	status: Current loading status of the file indicated by FileLink. This element MUST be set to one of the following values:	
FILE.058	B.1.13	statusTime: This element MUST be set to the time at which file status transitioned to the value indicated in the status element.	
GEN.001	4.1	IEEE 2030.5 servers and clients SHALL be compliant with [RFC 7230], [RFC 7231], [RFC 7232], [RFC 21 7233], [RFC 7234], and [RFC 7235].	All tests
GEN.002	4.2	The TCP ports used for HTTP or HTTPS SHALL be specified in the xmDNS service advertisement for the service.	COMM.001
GEN.003	4.2	Content SHALL be transferred with either one of the content types: ""application/sep+xml"" or ""application/sep-exi"".	All tests
GEN.004	4.2	All resources SHALL contain links to their subordinate resources to support flexibility in URIs and future extensibility.	All tests
GEN.005	4.2	Hexadecimal values SHALL be represented with one leading zero, if needed, to ensure an even number of digits.	All tests
GEN.006	4.3	IEEE 2030.5 devices SHALL conform to the interface specifications contained in the WADL.	All tests
GEN.007	4.3	* Devices SHALL conform to the WADL specification as per [WADL].	All tests
GEN.008	4.3	* Devices SHALL conform to the WADL definition in [IEEE 2030.5 SM].	All tests
GEN.009	4.3	By implication, all resource representations SHALL validate per the schema [ZB 13-0201] within the standardized SEP XML namespace (http://zigbee.org/sep).	All tests
GEN.010	4.4	Resources located at URIs returned in the href attribute of ""Link"" specializations (e.g., EndDeviceListLink, SelfDeviceLink) SHALL conform to the schema definition for that object,	All tests
GEN.011	4.4	If a client PUTs or POSTs a resource to a server containing attributes or elements that instead are to be populated by the server (e.g., href), the server SHALL return an HTTP 400 error.	All tests
GEN.012	4.4	If a function set is not implemented, Link elements to resources in that function set SHALL NOT be included.	All tests
GEN.013	4.5	URIs SHALL NOT be greater than 255 bytes in length. In practice, URIs SHOULD be much smaller than 80 bytes.	All tests
GEN.014	4.6	A list resource's elements SHALL be ordered according to this specified List Ordering.	All tests
GEN.015	4.6.1	The first ordinal position of the list SHALL be designated with a value of '0' and the maximum possible value is '65535'.	All tests
GEN.016	4.6.1	If this query string parameter is not specified, the default start value SHALL be '0'.	All tests
GEN.017	4.6.1	The parameter SHALL be ignored if the primary key is not time-based.	All tests
GEN.018	4.6.1	The format of the parameter SHALL be a 64-bit decimal number with identical semantics as that of the TimeType (see Section 10.2 and the XML XSD in [ZB 13-0201]).	All tests
GEN.019	4.6.1	If this query string parameter is not specified, the default limit SHALL be '1'.	All tests
GEN.020	4.6.1	If both a ""start"" and ""after"" query string parameter are used simultaneously, the ""after"" query string parameter SHALL have precedence.	All tests
GEN.021	4.6.1	The ""start"" position 0 SHALL be relative to the position specified by the ""after"" parameter.	All tests
GEN.022	4.6.1	If a query string requests a list element that does not exist (e.g., s=3 when there are two items in the list), servers SHALL return an empty list representation.	All tests
GEN.023	4.6.1	If a particular query string parameter appears more than once, then the first occurrence of the query string parameter SHALL be used (in left-to-right order) and subsequent occurrences MUST be ignored.	All tests
GEN.024	4.6.1	Server receipt of a query parameter unknown to the server MUST be ignored by the server and MUST NOT generate an HTTP error.	All tests
GEN.025	4.6.1	Servers MUST NOT generate resource representations containing href attributes that contain query parameters.	All tests
GEN.026	4.6.1	Clients MUST ignore query parameters contained in resource hrefs, but SHOULD NOT remove them if the URI is used for subsequent RESTful exchanges.	All tests

Req. ID	Section	Description	Test IDs
GEN.027	4.6.1	Should an empty list representation be requested (either through the use of query string parameters such as l=0 or when the list itself is empty), the server SHALL return no subordinate representations, but SHALL return any other elements that may be defined for the list.	All tests
GEN.028	4.6.1	Clients MUST NOT assume any index semantics for list URIs.	All tests
GEN.029	4.6.1	For example, a client desiring to read the 3 rd item from the list at http://some-host/somelist MUST NOT assume a GET to http://some-host/somelist/2 will return the third item.	
GEN.030	4.7	List resources SHALL support ""start"" and ""limit"" query string parameters, thus always supporting paging.	All tests
GEN.031	4.7	List resources that have a time-based primary key SHALL support the ""after"" query string parameters.	All tests
GEN.032	4.7	All subordinate resources of list resources SHALL include an href attribute containing the URI of the subordinate resource.	All tests
GEN.033	4.7	When queried, list resources SHALL return subordinate resources in the order defined by the List Ordering.	All tests
GEN.034	4.7	All subordinate resources of list resources that support multiple types, e.g., NotificationList, SHALL include an xsi:type attribute.	All tests
GEN.035	4.7	In this case, the XML Schema Instance Namespace must also be declared.	
GEN.036	4.7	Non-list resources SHALL NOT support the defined query string parameters.	All tests
GEN.037	5.5	However, IEEE 2030.5 clients may encounter any HTTP response code defined by [RFC 2616] and, all use of, and response to HTTP response codes SHALL be specification and RFC compliant.	All tests
GEN.038	5.5.1.1	A client MUST be prepared to accept one or more 1xx status responses prior to a regular response, even if the client does not expect a 100 (Continue) status message.	All tests
GEN.039	5.5.1.3	The Location header SHALL be used in conjunction with this response code to indicate the URI of the newly created resource.	All tests
GEN.040	5.5.1.3	If a new resource is created, the origin server MUST inform the user agent via the 201 (Created) response.	All tests
GEN.041	5.5.1.5	Note that [RFC 2616] requires the Content-Range and Date headers MUST be present in the response.	All tests
GEN.042	5.5.1.7	The Location header SHALL be used in conjunction with this response code to indicate the new URI of the requested resource.	All tests
GEN.043	5.5.1.8	All Internet-based HTTP/1.1 servers MUST respond with a 400 (Bad Request) status code to any HTTP/1.1 request message which lacks a Host header field.	All tests
GEN.044	5.5.1.9	The response MUST include a WWW-Authenticated header field containing a challenge applicable to the requested resource.	All tests
GEN.045	5.5.1.11	The response MUST include an Allow header containing a list of valid methods for the requested resource.	All tests
GEN.046	5.5.1.14	A response with status code 206 (Partial Content) MUST NOT include a Content-Range field with a byte-range-resp-spec of ""*"".	All tests
GEN.047	5.5.1.15	If a server receives a request containing an Expect field that includes an expectation-extension that it does not support, it MUST respond with a 417 (Expectation Failed) status.	All tests
GEN.048	5.5.1.17	The recipient of the entity MUST NOT ignore any Content - (e.g., Content-Range) headers that it does not understand or implement and MUST return a 501 (Not Implemented) response in such cases.	All tests
GEN.049	5.5.2	Should a client wish to operate with minimal understanding of HTTP response codes, it need only examine the first digit of the response code to understand the general category of the response and ""treat any unrecognized response as being equivalent to the x00 status code of that class, with the exception that an unrecognized response MUST NOT be cached"".	All tests
GEN.050	5.6	Application payload message encoding using both XML [XML] and EXI [EXI] SHALL be supported by all servers.	All tests
GEN.051	5.6	Application payload message encoding using either XML [XML] or EXI [EXI] SHALL be supported by all clients.	All tests
GEN.052	5.6.1	The XML version used SHALL be 1.0.	All tests
GEN.053	5.6.1	For XML payloads, the encoding SHALL be UTF-8.	All tests

Req. ID	Section	Description	Test IDs
GEN.054	5.6.2	The options for encoding EXI documents SHALL be as follows, and transactions will likely fail if different options are declared in the EXI option header.	
GEN.055	5.6.2	The following XML document describes the EXI option header that SHALL be used to encode the messages.	
GEN.056	5.7	A client SHALL declare acceptable media types using the HTTP Accept header.	COMM-003
GEN.057	5.7.1	That is, the price of any (TOU Tier N, Consumption Block M) SHALL be less than or equal to the price of (TOU Tier Nplus-one, Consumption Block M).	
GEN.058	5.7.1	Devices SHALL NOT send messages to nodes that declare ""-S0"" using arbitrary types, elements, and attributes.	
GEN.059	5.7.1	The grammar used for EXI SHALL be generated as a non-strict grammar only, as having both strict and non-strict grammars would put a large burden on storage requirements for certain devices.	
GEN.060	11.2	Proprietary extensions SHALL NOT be made using the ""smartenergy"" Service Type.	
GEN.061	11.4	Proprietary extensions SHALL NOT place any objects, elements, attributes, etc. in the standardized SEP XML namespace (""http://zigbee.org/sep"") and instead SHALL be placed in a different XML namespace.	
GEN.063	11.4	Proprietary extensions SHALL NOT extend enumerations defined in the IEEE 2030.5 schema.	
GEN.064	11.4	Proprietary extensions made to standardized objects (in a proprietary namespace) defined in the IEEE 2030.5 schema SHALL be able to be ignored.	
GEN.065	11.4	Proprietary extensions made to standardized objects (in a proprietary namespace) defined in the IEEE 2030.5 schema SHALL NOT change the semantics of elements and attributes defined in the IEEE 2030.5 schema.	
GEN.066	11.5	Should a proprietary extension wish to use the standard DeviceCapabilities resource, it MUST do so following the same rules as for other resources.	
LINK.001	B.1.22	AccountBalanceLink SHALL contain a Link to an instance of AccountBalance.	
LINK.002	B.1.22	ActiveBillingPeriodListLink SHALL contain a Link to a List of active BillingPeriod instances.	
LINK.003	B.1.22	ActiveCreditRegisterListLink SHALL contain a Link to a List of active CreditRegister instances.	
LINK.004	B.1.22	ActiveDERControlListLink SHALL contain a Link to a List of active DERControl instances.	
LINK.005	B.1.22	ActiveEndDeviceControlListLink SHALL contain a Link to a List of active EndDeviceControl instances.	
LINK.006	B.1.22	ActiveFlowReservationListLink SHALL contain a Link to a List of active FlowReservation instances.	
LINK.007	B.1.22	ActiveProjectionReadingListLink SHALL contain a Link to a List of active ProjectionReading instances.	
LINK.008	B.1.22	ActiveSupplyInterruptionOverrideListLink SHALL contain a Link to a List of active SupplyInterruptionOverride instances.	
LINK.009	B.1.22	ActiveTargetReadingListLink SHALL contain a Link to a List of active TargetReading instances.	
LINK.010	B.1.22	ActiveTextMessageListLink SHALL contain a Link to a List of active TextMessage instances.	
LINK.011	B.1.22	ActiveTimeTariffIntervalListLink SHALL contain a Link to a List of active TimeTariffInterval instances.	
LINK.012	B.1.22	AssociatedDERProgramListLink SHALL contain a Link to a List of DERPrograms having the DERControl(s) for this DER.	
LINK.013	B.1.22	UsagePointLink SHALL contain a Link to an instance of UsagePoint.	
LINK.014	B.1.22	BillingPeriodListLink SHALL contain a Link to a List of BillingPeriod instances.	
LINK.015	B.1.22	BillingReadingListLink SHALL contain a Link to a List of BillingReading instances.	
LINK.016	B.1.22	BillingReadingSetListLink SHALL contain a Link to a List of BillingReadingSet instances.	
LINK.017	B.1.22	ConfigurationLink SHALL contain a Link to an instance of Configuration.	
LINK.018	B.1.22	ConsumptionTariffIntervalListLink SHALL contain a Link to a List of ConsumptionTariffInterval instances.	

Req. ID	Section	Description	Test IDs
LINK.019	B.1.22	CreditRegisterListLink SHALL contain a Link to a List of CreditRegister instances.	
LINK.020	B.1.22	CustomerAccountLink SHALL contain a Link to an instance of CustomerAccount.	
LINK.021	B.1.22	CustomerAccountListLink SHALL contain a Link to a List of CustomerAccount instances.	
LINK.022	B.1.22	CustomerAgreementListLink SHALL contain a Link to a List of CustomerAgreement instances.	
LINK.023	B.1.22	DemandResponseProgramLink SHALL contain a Link to an instance of DemandResponseProgram.	
LINK.024	B.1.22	DemandResponseProgramListLink SHALL contain a Link to a List of DemandResponseProgram instances.	
LINK.025	B.1.22	DERAvailabilityLink SHALL contain a Link to an instance of DERAvailability.	
LINK.026	B.1.22	DERCapabilityLink SHALL contain a Link to an instance of DERCapability.	
LINK.027	B.1.22	DefaultDERControlLink SHALL contain a Link to an instance of DefaultDERControl.	
LINK.028	B.1.22	DERControlListLink SHALL contain a Link to a List of DERControl instances.	
LINK.029	B.1.22	DERCurveLink SHALL contain a Link to an instance of DERCurve.	
LINK.030	B.1.22	DERCurveListLink SHALL contain a Link to a List of DERCurve instances.	
LINK.031	B.1.22	DERLink SHALL contain a Link to an instance of DER.	
LINK.032	B.1.22	DERListLink SHALL contain a Link to a List of DER instances.	
LINK.033	B.1.22	DERProgramLink SHALL contain a Link to an instance of DERProgram.	
LINK.034	B.1.22	DERProgramListLink SHALL contain a Link to a List of DERProgram instances.	
LINK.035	B.1.22	DERSettingsLink SHALL contain a Link to an instance of DERSettings.	
LINK.036	B.1.22	DERStatusLink SHALL contain a Link to an instance of DERStatus.	
LINK.037	B.1.22	DeviceCapabilityLink SHALL contain a Link to an instance of DeviceCapability.	
LINK.038	B.1.22	DeviceInformationLink SHALL contain a Link to an instance of DeviceInformation.	
LINK.039	B.1.22	DeviceStatusLink SHALL contain a Link to an instance of DeviceStatus.	
LINK.040	B.1.22	EndDeviceControlListLink SHALL contain a Link to a List of EndDeviceControl instances.	
LINK.041	B.1.22	EndDeviceLink SHALL contain a Link to an instance of EndDevice.	
LINK.042	B.1.22	EndDeviceListLink SHALL contain a Link to a List of EndDevice instances.	
LINK.043	B.1.22	FileLink: This element MUST be set to the URI of the most recent File being loaded/activated by the LD.	
LINK.044	B.1.22	FileLink: In the case of file status 0, this element MUST be omitted.	
LINK.045	B.1.22	FileListLink SHALL contain a Link to a List of File instances.	
LINK.046	B.1.22	FileStatusLink SHALL contain a Link to an instance of FileStatus.	
LINK.047	B.1.22	FlowReservationRequestListLink SHALL contain a Link to a List of FlowReservationRequest instances.	
LINK.048	B.1.22	FlowReservationResponseListLink SHALL contain a Link to a List of FlowReservationResponse instances.	
LINK.049	B.1.22	FunctionSetAssignmentsListLink SHALL contain a Link to a List of FunctionSetAssignments instances.	
LINK.050	B.1.22	HistoricalReadingListLink SHALL contain a Link to a List of HistoricalReading instances.	
LINK.051	B.1.22	IPAddrListLink SHALL contain a Link to a List of IPAddr instances.	
LINK.052	B.1.22	IPInterfaceListLink SHALL contain a Link to a List of IPInterface instances.	
LINK.053	B.1.22	LLInterfaceListLink SHALL contain a Link to a List of LLInterface instances.	
LINK.054	B.1.22	SHALL contain a Link to a List of LoadShedAvailability instances.	
LINK.055	B.1.22	LogEventListLink SHALL contain a Link to a List of LogEvent instances.	
LINK.056	B.1.22	MessagingProgramListLink SHALL contain a Link to a List of MessagingProgram instances.	

Req. ID	Section	Description	Test IDs
LINK.057	B.1.22	MeterReadingLink SHALL contain a Link to an instance of MeterReading.	
LINK.058	B.1.22	MeterReadingListLink SHALL contain a Link to a List of MeterReading instances.	
LINK.059	B.1.22	MirrorUsagePointListLink SHALL contain a Link to a List of MirrorUsagePoint instances.	
LINK.060	B.1.22	NeighborListLink SHALL contain a Link to a List of Neighbor instances.	
LINK.061	B.1.22	NotificationListLink SHALL contain a Link to a List of Notification instances.	
LINK.062	B.1.22	PowerStatusLink SHALL contain a Link to an instance of PowerStatus.	
LINK.063	B.1.22	PrepaymentLink SHALL contain a Link to an instance of Prepayment.	
LINK.064	B.1.22	PrepaymentListLink SHALL contain a Link to a List of Prepayment instances.	
LINK.065	B.1.22	PrepayOperationStatusLink SHALL contain a Link to an instance of PrepayOperationStatus.	
LINK.066	B.1.22	PriceResponseCfgListLink SHALL contain a Link to a List of PriceResponseCfg instances.	
LINK.067	B.1.22	ProjectionReadingListLink SHALL contain a Link to a List of ProjectionReading instances.	
LINK.068	B.1.22	RateComponentLink SHALL contain a Link to an instance of RateComponent.	
LINK.069	B.1.22	RateComponentListLink SHALL contain a Link to a List of RateComponent instances.	
LINK.070	B.1.22	ReadingListLink SHALL contain a Link to a List of Reading instances.	
LINK.071	B.1.22	ReadingSetListLink SHALL contain a Link to a List of ReadingSet instances.	
LINK.072	B.1.22	ReadingTypeLink SHALL contain a Link to an instance of ReadingType.	
LINK.073	B.1.22	RegistrationLink SHALL contain a Link to an instance of Registration.	
LINK.074	B.1.22	ResponseListLink SHALL contain a Link to a List of Response instances.	
LINK.075	B.1.22	ResponseSetListLink SHALL contain a Link to a List of ResponseSet instances.	
LINK.076	B.1.22	RPLInstanceListLink SHALL contain a Link to a List of RPLInterface instances.	
LINK.077	B.1.22	RPLSourceRoutesListLink SHALL contain a Link to a List of RPLSourceRoutes instances.	
LINK.078	B.1.22	SelfDeviceLink SHALL contain a Link to an instance of SelfDevice.	
LINK.079	B.1.22	ServiceSupplierLink SHALL contain a Link to an instance of ServiceSupplier.	
LINK.080	B.1.22	SubscriptionListLink SHALL contain a Link to a List of Subscription instances.	
LINK.081	B.1.22	SupplyInterruptionOverrideListLink SHALL contain a Link to a List of SupplyInterruptionOverride instances.	
LINK.082	B.1.22	SupportedLocaleListLink SHALL contain a Link to a List of SupportedLocale instances.	
LINK.083	B.1.22	TargetReadingListLink SHALL contain a Link to a List of TargetReading instances.	
LINK.084	B.1.22	TariffProfileLink SHALL contain a Link to an instance of TariffProfile.	
LINK.085	B.1.22	TariffProfileListLink SHALL contain a Link to a List of TariffProfile instances.	
LINK.086	B.1.22	TextMessageListLink SHALL contain a Link to a List of TextMessage instances.	
LINK.087	B.1.22	TimeLink SHALL contain a Link to an instance of Time.	
LINK.088	B.1.22	TimeTariffIntervalListLink SHALL contain a Link to a List of TimeTariffInterval instances.	
LINK.090	B.1.22	UsagePointListLink SHALL contain a Link to a List of UsagePoint instances.	
LOG.001	9.5.3	The number of LogEvents supported within the LogEvent List SHALL be vendor dependent.	BASIC-027
LOG.002	9.5.3	LogEvent server devices SHALL be capable of internally storing and supporting at least 10 unique LogEvent instances.	BASIC-027
LOG.003	9.5.3	When a new LogEvent is issued, it SHALL be added as the first entry in the LogEvent List.	BASIC-027
LOG.004	9.5.3	LogEvents SHALL be time stamped when they are added to the LogEventList.	BASIC-027
LOG.005	9.5.3	A LogEvent SHALL be defined by this standard or vendor defined.[SR(EC54].	BASIC-027

Req. ID	Section	Description	Test IDs
LOG.006	B.1.11	If the profileID indicates this is IEEE 2030.5, the functionSet is defined by IEEE 2030.5 and SHALL be one of the values from the table below (IEEE 2030.5 function set identifiers).	BASIC-027
LOG.007	B.1.11	ZigBee-assigned logEventCodes SHALL use the ZigBee Alliance PEN.	
LOG.008	B.1.11	Combinations of profileID, functionSet, and logEventCode SHALL have unique meaning within a logEventPEN and are defined by the owner of the PEN.	BASIC-027
METER.001	10.3.2	While a ReadingSet is in this state, ReadingSet.timePeriod.start SHALL be when the ReadingSet starts recording its first value and that ReadingSet.timePeriod.duration SHALL grow each time the ReadingSet is updated.	
METER.002	10.3.2	The ReadingSet.mRID field SHALL be assigned a value of 0xFFFFFFFFFFFFFFFFFFFFFFFF[XXXXXXXX] (Where [XXXXXXXX] is replaced by the manufacturer's PEN) while the data is being recorded and changed to an appropriate mRID when the ReadingSet is complete.	
METER.003	10.3.2	A Metering function set instance that provides instantaneous demand data SHALL serve a Meter Reading resource (and subordinate resources) with the following properties:	
METER.004	10.3.2	* SHALL contain a ReadingTypeLink element that points to a ReadingType resource that matches the InstantaneousDemand definition from Table 40.	
METER.005	10.3.2	* SHALL contain a ReadingLink element that points to a Reading resource that contains the instantaneous demand value.	
METER.006	10.3.2	* SHALL NOT contain a ReadingSetListLink element.	
METER.007	10.3.2	A Metering function set instance that provides summation delivered data SHALL serve a Meter Reading resource (and subordinate resources) with the following properties:	
METER.008	10.3.2	* SHALL contain a ReadingTypeLink element that points to a ReadingType resource that matches the SummationDelivered definition from Table 40.	
METER.009	10.3.2	The ReadingType resource SHALL specify the intervalLength that is the default for the intervals contained in the ReadingList resource.	
METER.010	10.3.2	* SHALL contain a ReadingSetListLink element that points to a ReadingSetList resource.	
METER.011	10.3.2	When metering data is present, the ReadingSetList SHALL contain at least one ReadingSet resource, which corresponds to the present Interval Block data (the one currently being filled).	
METER.012	10.3.2	* The ReadingSet resource SHALL contain a ReadingListLink element that points to a ReadingList resource.	
METER.013	10.3.2	The ReadingList resource SHALL contain (number of TOU tiers + 1) multiplied by (number of consumption blocks + 1) Reading resources.	
METER.014	10.3.2	The Reading resources in the ReadingList SHALL correspond to the summation delivered value for each combination of consumptionBlock=0..(number of consumption blocks) and touTier=0..(number of TOU tiers).	
METER.015	10.3.2	The Reading for (consumptionBlock=0, touTier=0) SHALL correspond to the total summation delivered.	
METER.016	10.3.2	The Reading for (consumptionBlock=x >0, touTier=0) SHALL correspond to the Block x summation delivered (across all TOU tiers).	
METER.017	10.3.2	The Reading for (consumptionBlock=0, touTier=y >0) SHALL correspond to the TOU Tier y summation delivered (across all consumption blocks).	
METER.018	10.3.2	The Reading for (consumptionBlock=x > 0, touTier=y >0) SHALL correspond to the consumptionBlock x, touTier y summation delivered.	
METER.019	10.3.2	* SHALL contain a ReadingLink to a Reading resource that contains the present summation delivered, which is semantically equivalent to the Reading for (consumptionBlock=0, touTier=0) for the present ReadingList.	
METER.020	10.3.2	A Metering function set instance that provides summation received data, maximum demand delivered data, maximum demand received data, and / or other reading type data that utilizes TOU tiers and / or consumption blocks SHALL serve a Meter Reading resource (and subordinate resources) as per the rules for Summation Delivered above, but with the ReadingType resource matching the appropriate definition in Table 12-3, and with the Reading values corresponding to the appropriate data type.	
METER.021	10.3.2	A Metering function set instance that provides interval data SHALL serve a Meter Reading resource (and subordinate resources) with the following properties:	
METER.022	10.3.2	* SHALL contain a ReadingTypeLink element that points to a ReadingType resource that matches an Interval data definition from Table 40.	

Req. ID	Section	Description	Test IDs
METER.023	10.3.2	Watt-PowerFactor curve types SHALL specify two dependent variables (yvalue plus excitation) per curveData point.	
METER.024	10.3.2	* SHALL contain a ReadingSetListLink element that points to a ReadingSetList resource.	
METER.025	10.3.2	o When metering data is present, the ReadingSetList SHALL contain at least one ReadingSet source, which MAY correspond[SR(EC74] to the present Interval Block data (the one currently being filled).	
METER.027	10.3.2	The ReadingList resource SHALL contain Reading resources each of which represents a portion (interval) of the timePeriod of the ReadingSet.	
METER.028	10.3.2	If the duration in the Time Period of the Reading is not equal to the intervalLength specified in the ReadingType the timePeriod SHALL be included in the Reading.	
METER.029	10.3.2	* SHALL NOT contain a ReadingLink resource.	
METER.030	10.3.2	A metering instance must set these values as appropriate for its commodities.	
METER.031	B.1.15	localID attribute: For interval data, this value SHALL increase with each interval time, and for block/tier readings, localID SHALL not be specified.	
METER.032	B.1.15	numberOfTouTiers: Servers SHALL populate this value with the largest touTier value that will ever be used while this ReadingType is in effect.	
METER.033	B.1.15	numberOfTouTiers: Servers SHALL set numberOfTouTiers equal to the number of standard TOU tiers plus the number of CPP tiers that may be used while this ReadingType is in effect.	
METER.034	B.1.15	numberOfTouTiers: Servers SHALL specify a value between 1 and 255 (inclusive) for numberOfTouTiers (servers providing flat rate pricing SHALL set numberOfTouTiers to 1, as in practice there is no difference between having no tiers and having one tier).	
METER.035	B.1.15	subIntervalLength: It SHALL be an integral division of the intervalLength.	
METER.036	B.1.15	This attribute SHALL be present when mirroring.	
MULTI.001	10.1.5.2	Devices MUST complete the registration and service discovery process for each of the function set servers with which the user intends it to access information.	COMM-003
MULTI.002	10.1.5.3	Each function set server that has a reference to time SHALL also serve its respective time to the HAN.	
MULTI.003	10.1.5.3	Clients SHALL choose a primary time server in the HAN with which to align its internal time per the Time function set guidelines.	
MULTI.004	10.1.5.6	Clients SHALL only report acting on one Event at a time and SHALL NOT respond to multiple DRLC or DER servers that they are acting on multiple Events for that function set simultaneously.	
MULTI.005	10.1.5.6	If clients are getting Events from multiple servers of the same function set they SHALL detect duplicate Events by comparing the mRIDs of the Events.	
MULTI.006	10.1.5.6	When devices are registered to one or more DRLC servers, they SHALL NOT act upon any public DRLC servers that are present in the HAN or become available.	
MULTI.007	10.1.5.6	When devices are registered to one or more DER Control servers, they SHALL NOT act upon any public DER Control servers that are present in the HAN or become available.	
MULTI.008	10.1.5.6	Clients SHALL determine the primacy of DRLC, and DER Control based on the following in order of precedence:	
MULTI.009	10.1.5.6	Servers SHALL indicate their primacy in the primacy element of the function set instance. See schema [ZB 13-0201] definition of PrimacyType for possible values.	
MULTI.010	10.1.5.6	Clients SHALL prioritize execution of DRLC and DER function set Events with different PrimacyType attributes using the following guidelines:	
MULTI.011	10.1.5.6	If a client is executing an Event from one server and decides to execute an Event from a different server per the application guidelines and if the superseded Event responseRequired indicates, the client SHALL send a response with ""Aborted due to an Alternate Server Event"" Response.status for the Event that was superseded.	
MULTI.012	10.1.5.6	Likewise, if a client is executing an Event from one program and decides to execute an Event from a different program on the same server per the application guidelines and if the superseded Event responseRequired indicates, the client SHALL send a response with ""Aborted due to an Alternate Program Event"" Response.status for the Event that was superseded.[SR(P68].	
MUP.001	10.10.3	A Metering Mirror function set server SHALL NOT advertise support for mirroring unless it has the resources available to host at least one additional mirror.	CORE-020, BASIC0207, BASIC-029

Req. ID	Section	Description	Test IDs
MUP.002	10.10.3	The server must have room for at least one instance of each of the resources possible under a Usage Point.	CORE-020, BASIC0207, BASIC-029
MUP.003	10.10.3	To create a new Metering Mirror the client SHALL POST to the server MirrorUsagePointList (e.g., /mup) for the mirrored usage point.	CORE-020, BASIC0207, BASIC-029
MUP.004	10.10.3	This POST SHALL contain at least the information through the definition of MirrorMeterReadings and ReadingType including the MirrorUsagePoint mRID and MirrorMeterReading mRIDs.	CORE-020, BASIC0207, BASIC-029
MUP.005	10.10.3	If the mRID of the MirrorUsagePoint is unique (does not match a 4108 MirrorUsagePoint.mRID of an existing MirrorUsagePoint) the response SHALL be response code 201 (Created), the MirrorUsagePoint URI SHALL be included in the Location header.	CORE-020, BASIC0207, BASIC-029
MUP.006	10.10.3	If the mRID of the MirrorUsagePoint matches an existing MirrorUsagePoint, the new data SHALL be written over the existing MirrorUsagePoint (and associated UsagePoint) and the response code SHALL be 204 (No Content), the MirrorUsagePoint URI SHALL be included in the Location header. If the MirrorUsagePoint contains MirrorMeterReadings, then the guidance of rules 8 and 9 are to be applied.	CORE-020, BASIC0207, BASIC-029
MUP.007	10.10.3	When the Metering Mirror function set server receives a POST, it SHALL copy the received data, including mRIDs, into the normal metering structure to its Metering UsagePoint structure (e.g., /upt), and it SHALL allocate enough resources to manage the mirror and its data.	CORE-020, BASIC0207, BASIC-029
MUP.008	10.10.3	A GET of the resource (MirrorUsagePoint) identified in the response to the initial POST SHALL return a resource with only the first level elements (i.e., sub-elements and collections are not included).	CORE-020, BASIC0207, BASIC-029
MUP.009	10.10.3	To POST new data to an existing MirrorUsagePoint, the Metering client SHALL POST a MirrorMeterReading or MirrorMeterReadingList containing MirrorReadingSets and/or Readings to the resource identified in the Metering server response to the POST that created the resource (e.g., /mup/3).	CORE-020, BASIC0207, BASIC-029
MUP.010	10.10.3	If a POST to the MirrorUsagePoint is of a MirrorMeterReading then a successful response SHALL contain a Location header indicating the URI of the MeterReading resource under the associated UsagePoint (e.g., /upt/2/mr/3).	CORE-020, BASIC0207, BASIC-029
MUP.011	10.10.3	If a POST to the MirrorUsagePoint is of a MirrorMeterReadingList then a successful response SHALL contain a Location header indicating the URI of the MeterReadingList under the associated UsagePoint (e.g., /upt/2/mr).	CORE-020, BASIC0207, BASIC-029
MUP.012	10.10.3	In a POST to the MirrorUsagePoint, the mRID attribute of the MirrorMeterReading(s) SHALL be used by the Metering Mirror server to associate the data in a POST with the MeterReading in the associated UsagePoint.	CORE-020, BASIC0207, BASIC-029
MUP.013	10.10.3	In a POST to the MirrorUsagePoint, if the mRID attribute matches a previous MirrorMeterReading then the contained MirrorReadingSets SHALL be added to the associated MeterReading and a contained Reading SHALL replace the existing Reading.	CORE-020, BASIC0207, BASIC-029
MUP.014	10.10.3	The contents of the MirrorMeterReading SHALL overwrite the data in the associated MeterReading.[SR(EC99].	CORE-020, BASIC-029
MUP.015	10.10.3	In a POST to the MirrorUsagePoint, if the mRID does not match a previous MirrorMeterReading and it contains a ReadingType, a new MeterReading SHALL be created under the associated UsagePoint with the new data.	CORE-020, BASIC-029
MUP.016	10.10.3	In a POST to the MirrorUsagePoint, if the mRID does not match a previous MirrorMeterReading and there is not a ReadingType then the request SHALL be rejected with a response code 400 (Bad Request).	CORE-020, BASIC-029
MUP.017	10.10.3	In a POST to the MirrorUsagePoint, where the request is not rejected, the new data SHALL be applied to the related UsagePoint resource structure according to the following:	CORE-020, BASIC-029
MUP.018	10.10.3	If a MirrorReadingSet is received with a duplicate mRID of an existing ReadingSet, and it is targeted within the same resource hierarchy, then the new data SHALL replace the existing data of the identified ReadingSet.	CORE-020, BASIC-029
MUP.019	10.10.3	If a MirrorReadingSet is received with a unique mRID then the new data SHALL be added to the identified ReadingSetList.	CORE-020, BASIC-029
MUP.020	10.10.3	If a client POSTs more data than the Metering Mirror server is willing to accept, the server SHALL respond with a response code of 413 (Request Entity Too Large).	
MUP.021	10.10.3	When a MirrorUsagePoint is deleted, the associated UsagePoint SHALL also be deleted.[SR(EC100][GL101].	

Req. ID	Section	Description	Test IDs
NSTAT.001	B.1.10	EUI64 attribute (HexBinary64) Contains the EUI-64 of the link layer interface. 48-bit MAC addresses SHALL be changed into an EUI-64 using the method defined in [RFC 4291], Appendix A.	
QLI.001		Clients SHALL send Response messages within the specified timeframe as required by the event.	All event tests that require Responses
QLI.002		Clients SHALL poll the resources of this function set (and all resources below) every pollRate seconds.	Core-003
RAND.001	10.1.4.2	Any device handling these Events and not operating on them SHALL NOT modify or apply them.	CORE-021
RAND.002	10.1.4.2.1	The device SHALL randomly select a number in seconds from zero to the randomizeStart specified for this event.	CORE-021
RAND.003	10.1.4.2.1	Depending on the sign of the value (positive or negative), randomization SHALL be applied before or after the scheduled Start Time of the Event.	CORE-021
RAND.004	10.1.4.2.1	If the value is negative, randomization SHALL be applied before the specified Start Time of the Event.	CORE-021
RAND.005	10.1.4.2.1	If the value is positive, randomization SHALL be applied after the specified Start Time of the event, causing the device to delay commencement of the Event.	CORE-021
RAND.006	10.1.4.2.2	Devices SHALL randomly select a number in seconds from zero to the randomizeDuration value specified for the Event.	CORE-021
RAND.007	10.1.4.2.2	Depending on the sign of the value (positive or negative), randomization SHALL be applied to increase or decrease the duration of the event.	CORE-021
RAND.008	10.1.4.2.2	If the value is negative, randomization SHALL decrease the EffectiveDuration of the Event.	CORE-021
RAND.009	10.1.4.2.2	If the value is positive, randomization SHALL be added to the duration of the event, causing the device to delay termination of the Event.	CORE-021
RAND.010	10.1.4.3	Clients that use fixed pseudo random values SHALL scale the applied randomization based on the range indicated by the given randomizeStart or randomizeDuration.	CORE-021
RAND.011	10.1.4.3	Stated differently, fixed pseudo random values SHALL indicate a percentage of the randomizeStart or randomizeDuration to be applied.	CORE-021
RAND.012	10.1.4.3	Manufacturers SHALL assure that fixed pseudo randomization values are random over a population of like devices.	CORE-021
RAND.013	10.1.4.3	Manufacturers SHALL assure that random number generators maintain a distribution of values over a population of like devices (e.g., by using differing seeds).[SR(EC65].	CORE-021
RAND.014	10.1.4.3	Clients that act upon Events SHALL support randomization.	CORE-021
RAND.015	10.1.4.3	Clients SHALL apply randomization attributes when present in an Event's attributes.	CORE-021
RAND.016	10.1.4.3	Cancellation of an active Event SHALL cause clients to apply the greater of (absolute value of randomizeStart attribute) and (absolute value of randomizeDuration) to the abbreviated Event.	CORE-021
RAND.017	10.1.4.3	If an Event is in progress and user override occurs, the client SHALL respond to the user override without randomization.	CORE-021
SEC.001	6.3.2	The SFDI SHALL be the certificate fingerprint left-truncated to 36-bits.	COMM-006, BASIC-001, UTIL-001
SEC.002	6.3.2	For display purposes, this SHALL be expressed as 11 decimal (base 10) digits, with an additional sum-of-digits checksum digit right-concatenated.	COMM-006, BASIC-001, UTIL-001
SEC.003	6.3.2	For input validation purposes, the sum of the digits of the fingerprint including the checksum digit, modulo 10, SHALL be zero.	COMM-006, BASIC-001, UTIL-001
SEC.004	6.3.3	The LFDI SHALL be the certificate fingerprint left-truncated to 160-bits (20 octets).	COMM-006, BASIC-001, UTIL-001
SEC.005	6.3.3	For display purposes, this SHALL be expressed as 40 hexadecimal (base 16) digits in groups of four.	COMM-006, BASIC-001, UTIL-001
SEC.006	6.3.4	For display purposes, this SHALL be expressed as 5 decimal (base 10) digits, with an additional sum-of-digits checksum digit right-concatenated:	COMM-006, BASIC-001, UTIL-001
SEC.007	6.3.4	For input validation purposes, the sum of the digits of the PIN including the checksum digit, modulo 10, SHALL be zero.	COMM-006, BASIC-001, UTIL-001
SEC.008	6.3.4	The PIN is not overly secure and therefore SHALL NOT be used in any way to derive keys for actual data encryption.	COMM-006, BASIC-001, UTIL-001

Req. ID	Section	Description	Test IDs
SEC.009	6.4	Resource access requiring application layer authentication, data confidentiality and integrity checking SHALL occur through requests from a client to the server using HTTP over TLS [RFC 2818] (also known as HTTPS) using TLS version 1.2 [RFC 5246].	All Tests
SEC.010	6.4	Resource access not requiring application layer authentication, data confidentiality or integrity checking SHALL occur through requests from a client to the host server using HTTP [RFC 2616].	
SEC.011	6.4	If a request is made to the port associated with HTTPS, it is considered an HTTPS request and authentication SHALL have taken place.	
SEC.012	6.4	If authentication has not taken place, authentication SHALL be initiated as described in Section 6.5.	
SEC.013	6.4	If the request is made to the port associated with HTTP, it is considered an HTTP request and authentication SHALL NOT be REQUIRED and the request is then passed to the ACL associated with the resource.	COMM-005
SEC.014	6.5	The use of TLS [RFC 5246] requires that all hosts implementing server functionality SHALL use a Device Certificate whereby the server presents its Device Certificate as part of the TLS handshake.	All Tests
SEC.015	6.5	If no TLS session is in place, a TLS handshake SHALL occur between the client and server:.	COMM-003
SEC.016	6.5	Authentication of the server SHALL be done as part of the TLS handshake by validating its Device Certificate as described in [RFC 5246], Section 7 using the inherent PKI.	COMM-003
SEC.017	6.5	If the client has a Device Certificate, authentication of the client SHALL be done as part of the TLS handshake by validating the client's Device Certificate as described in [RFC 5246], Section 7 using the inherent PKI.	COMM-003
SEC.018	6.5	If the client has a Self-signed Certificate, the Self-signed Certificate SHALL be validated for correctness.	
SEC.019	6.6	If the client uses a Self-signed Certificate, pre-authorization using the SFDI of the Self-signed Certificate MUST have taken place and authorization SHALL be granted if the SFDI of the presented Self-signed Certificate matches the SFDI presented as part of registration.	
SEC.020	6.7	All devices SHALL support the TLS_ECDHE_ECDSA_WITH _AES_128_CCM_8 cipher suite [I-D AESCCM].	COMM-003
SEC.021	6.7	The ECC cipher suite SHALL use elliptic curve secp256r1.	COMM-003
SEC.022	6.7	All devices acting as a server SHALL support the ECC cipher suite.	COMM-003
SEC.023	6.7	In particular, all devices acting as a server SHALL have an ECC certificate.	COMM-003
SEC.024	6.7	All devices acting as a client SHALL support the ECC cipher suite for the purposes of validating an ECC certificate.	COMM-003
SEC.025	6.8	Servers SHALL be configurable to support each default policy for all implemented function sets during certification testing.	COMM-003
SEC.026	6.8	The Method value MUST contain GET (0x01).	COMM-003
SEC.027	6.8	Servers SHALL support the default policies for certification testing.	COMM-003
SEC.028	6.9	Registration for clients SHALL occur via an EndDevice resource corresponding to the client, which typically resides on an ESI associated with the utility, premises owner or third-party service provider that is trusted to perform registration.	COMM-003, CORE-007
SEC.029	6.9.1	Clients SHALL locate HAN services by performing DNS Service Discovery (DNS-SD) queries to the HAN, see Section 9 for details.	COMM-002
SEC.030	6.9.1	The EndDevice resource's server SHALL listen on the TCP port associated with HTTPS and follow the procedure described in [RFC 5246] when a client attempts to access the EndDevice resource.	BASIC-002, COMM-003, CORE-007, CORE-010
SEC.031	6.9.1	Authorization SHALL then occur whereby the ACLs of the server resources corresponding to the registering client are set according to the security policy and the presence in aclLocalRegistrationList.	COMM-003
SEC.032	6.9.1	If present, the client MUST verify that the EndDevice's associated Registration resource contains the correct PIN for the client.	COMM-003, CORE-007
SEC.033	6.9.1	The client SHALL subsequently re-use its Device Certificate to authenticate with any other server host. It does not need to re-authenticate with the EndDevice resource's server.	

Req. ID	Section	Description	Test IDs
SEC.034	6.1	The following attributes SHALL be exact matches in both servers' ReadingType instances:	
SEC.035	6.11.3	The IEEE 2030.5 Manufacturing PKI SHALL be a hierarchy with a depth of 2, 3 or 4 levels.	COMM-003, COMM-004
SEC.036	6.11.3	At the top level, Manufacturing PKI hierarchy SHALL have one SERCA.	
SEC.037	6.11.3	One or more SERCAs SHALL be nominated by the CSEP [CSEP] and SHALL be outsourced to a commercial CA service provider.	
SEC.038	6.11.3	Private key material for a SERCA MUST be held in a form to allow secure transfer of the material to a new commercial CA if necessary.	
SEC.039	6.11.3	The Manufacturing PKI hierarchy MAY include one IEEE 2030.5 MCA. One or more MCAs SHALL be out-sourced to an IEEE 2030.5 vendor, specifically for the issuance of vendor-specific MICAs.	
SEC.040	6.11.3	One or more MICAs SHALL be outsourced to a IEEE 2030.5 vendor, specifically for the production of IEEE 2030.5 certified devices by that vendor.	
SEC.041	6.11.3	All devices SHALL store exactly one SERCA in their certificate path.	COMM-004
SEC.042	6.11.3	All devices SHALL include at least the public keys of all existing SERCAs, and MAY include their certificates.	COMM-004
SEC.043	6.11.3.1	In such cases, the retired certificate and its associated private key SHALL no longer be used for issuing certificates.	
SEC.044	6.11.3.1	An MCA or MICA certificate SHALL NOT be re-issued (e.g., re-signed with the same SubjectName and public key, but with a new validity period).	
SEC.045	6.11.3.1	Instead, if it is desired to retire an existing key/certificate, a new key pair SHALL be generated and bound into a new MCA or MICA certificate generally with a new serialNumber component of the SubjectName.	
SEC.046	6.11.3.1	Retired certificates MUST remain available to verify subsidiary certificates.	
SEC.047	6.11.3.1	A side effect of the indefinite lifetime requirement coupled with the permanent embedding of the Device Certificate and its certificate chain within a device is that the device, MCA and MICA certificates MUST NOT and cannot be revoked once issued.	
SEC.048	6.11.3.1	Specifically, no MCA, MICA or SERCA shall issue or be required to issue CRLs and no MCA or MICA shall operate or have operated on their behalf any OCSP server for the purpose of providing validity information for any certificate under the Manufacturing PKI hierarchy.	
SEC.049	6.11.4.1	All certificates in the Manufacturing PKI SHALL be compliant with [RFC 5280].	COMM-004
SEC.050	6.11.4.2	Device Certificates and Device Test Certificates in the Manufacturing PKI SHALL be compliant with [IEEE 802.1AR] with the following exceptions:	COMM-004
SEC.051	6.11.4.2	Per [RFC 5280], this extension MUST be marked critical when the SubjectName field is empty.	COMM-004
SEC.052	6.11.5	The signature method for signatures formed by EC P-256 private keys MUST be SHA256 with ECDSA.	
SEC.053	6.11.5	Within the Manufacturing PKI hierarchy, all certificates MUST contain only an EC P-256 public key.	
SEC.054	6.11.5	That public key MUST contain an elliptic curve point in uncompressed form. See [RFC 5480] Section 2.2 for details on the uncompressed form.	
SEC.055	6.11.5	Per [RFC 5280], CAs MUST ensure the uniqueness of the serial numbers on the certificates they issue.	
SEC.056	6.11.5	Per [RFC 5280], the IssuerName of any certificate MUST be identical to the signer's SubjectName.	COMM-004
SEC.057	6.11.5	With the exception of Device Certificates and Device Test Certificates as described in Sections 6.11.2.3 and 6.11.2.4, the SubjectName MUST be non-empty.	COMM-004
SEC.058	6.11.6	The PolicyInformation object MUST NOT contain any policyQualifier fields.	COMM-004
SEC.059	6.11.6	Each certificate in the Manufacturing PKI hierarchy MUST have a Valid: notBefore field consisting of the time of issue encoded as per [RFC 5280] Section 4.1.2.5 and a Valid:notAfter field consisting of the GeneralizedTime value 99991231235959Z (see [RFC 5280], Section 4.1.2.5) for 256-bit ECC-based certificates.	COMM-004
SEC.060	6.11.6	Each CA certificate MUST contain a SubjectKeyIdentifier extension with an 8-octet key identifier generated as per method (2) of Section 4.2.1.2 of [RFC 5280].	COMM-004

Req. ID	Section	Description	Test IDs
SEC.061	6.11.6	A non-CA certificate MAY contain a SubjectKeyIdentifier extension - if it does, such extension MUST be generated as per method 2 of Section 4.2.1.2 of [RFC 5280].	COMM-004
SEC.062	6.11.6	In both cases, the extension MUST be marked non-critical.	
SEC.063	6.11.6	IEEE 2030.5 devices MUST be able to follow a chain where the key identifier was not generated in compliance with this section but where there is correspondence in actual values between a child AuthorityKeyIdentifier and a parent's (CA's) SubjectKeyIdentifier.	
SEC.064	6.11.6	Each certificate, except self-signed client certificates and root certificates, MUST contain an AuthorityKeyIdentifier extension of form [0] KeyIdentifier where the value of the KeyIdentifier field is taken from the value of the SubjectKeyIdentifier extension of the certificate issuer.	COMM-004
SEC.065	6.11.6	The extension MUST be marked non-critical.	
SEC.066	6.11.7.3	The OID MUST be unique for each different manufacturer's device model and / or type.	
SEC.067	6.11.7.3	The combination of hwType and hwSerialNum MUST be unique.	
SEC.068	6.11.8	IEEE 2030.5 devices MUST accept unexpected (not listed in this profile) certificate extensions and MUST silently ignore non-critical unrecognized certificate extensions.	
SEC.069	6.11.8	Per [RFC 5280], devices MUST reject any certificate containing unrecognized critical certificate extensions.	COMM-004
SEC.070	6.11.8.1	A SERCA instance (e.g., contracted for CA operation) SHALL have exactly one self-signed certificate.	
SEC.071	6.11.8.1	They MUST contain the extensions described below and MUST have the name form as described.	
SEC.072	6.11.8.2.1	MCA certificates MUST contain the extensions described below and MUST have at least the O and CN components of the Subject Name as described.	
SEC.073	6.11.8.2.2	MICA certificates MUST contain the extensions described below and MUST have at least the O and CN component of the Subject Name as described.	
SEC.074	6.11.8.2.3	Device Certificates MUST contain the extensions described below. The Subject Name field MUST be empty.	
SEC.075	6.11.8.2.3	The Subject Name field MUST be empty.	
SEC.076	6.11.8.2.3	The device type identifier OID(s) MUST be selected from those present in the issuing certificate's certificatePolicy extension.	
SEC.077	6.11.8.2.4	Device Test Certificates MUST contain the extensions described below.	
SEC.078	6.11.8.2.4	The Subject Name field MUST be empty.	
SEC.079	6.11.8.2.4	The device type identifier OID(s) MUST be selected from those present in the issuing certificate's certificatePolicy extension.	
SEC.080	6.11.8.3.1.2	The certificate installation for these additional certificates MUST include the complete chain of certificates needed to validate the certificate, including the root of trust certificate for the chain.	
SEC.081	6.11.8.3.1.2	The certificates MUST contain an RSA 2048-bit public key and MUST be signed using SHA256 with RSA.	
SEC.082	6.11.8.3.1.2	An RSA certificate MUST NOT use a SubjectAlternativeName extension in place of a SubjectName unless it is identical to the SubjectAlternativeName extension in the device's Device Certificate.	
SEC.083	6.11.8.3.1.3	If a IEEE 2030.5 device without an additional certificate receives a TLS Client Hello containing only an RSA cipher suites, it MUST reject the connection with the appropriate code.	
SEC.084	6.11.8.3.2	A IEEE 2030.5 device or application acting in a client role SHALL reject a self-signed certificate if presented by the server.	
SEC.085	6.11.8.3.2	The server SHALL verify that the Self-signed Certificate follows the format described below and SHALL reject the certificate if there are any discrepancies.	
SEC.086	6.11.8.3.2	A server MUST NOT treat a Self-signed Certificate received through a TLS Handshake as corresponding to a root CA, unless the public key carried in the Self-signed Certificate is the same as one of the pre- provisioned roots.	
SEC.087	6.11.8.3.4	For interoperability with this profile, those certificates MUST meet the following requirements.	
SEC.088	6.11.8.3.4	* MUST contain either an RSA 2048-bit public key or a EC P-256 public key.	

Req. ID	Section	Description	Test IDs
SEC.089	6.11.8.3.4	* MUST be signed either using either the SHA256withRSA or SHA256withECDSA algorithms.	
SEC.090	6.11.8.3.4	In addition, the IEEE 2030.5 client device MUST have a valid root of trust for the server certificate.	
SEC.091	6.11.9.2	Device manufacturers SHALL ensure that the current active Smart Energy Root CA (SERCA) certificates or root public keys are embedded in the device at the time of manufacture and firmware upgrade.	
SEC.092	6.11.9.2	They SHALL ensure that, once installed, the Root CA public keys cannot be overwritten via an over-the-air action, except as a by-product of a successful firmware upgrade.	
SEC.093	6.11.9.3	Device manufacturers SHALL ensure that a complete and valid Manufacturing PKI certificate chain (e.g., SERCA, MCA if any, issuing MICA if any, and Device Certificate) is embedded in the device at the time of manufacture.	
SEC.094	6.11.9.3	In an authentication exchange, the device SHALL supply a complete and valid chain comprising its own certificate and any intermediate CA certificate between the device and the root (i.e.; all certificates in its chain except its SERCA).	
SEC.095	6.11.9.4	A device meant to act as a server to non-IEEE 2030.5 entities MUST incorporate at manufacture a list of certificates for non-IEEE 2030.5 Root CAs recommended by [CSEP] for use as TLS trust anchors.	
SEC.096	6.11.9.4	The [CSEP] shall publish such a list of certificates on its website and shall update the list no less often than annually.	
SEC.097	6.11.9.4	There shall be an initial list size of 10 non-IEEE 2030.5 Root CA certificates, increasing by one a year to a maximum of 20 non-IEEE 2030.5 Root CA certificates.	
SEC.098	6.11.10	IEEE 2030.5 devices and applications MUST follow the procedure defined in [RFC 5280], Section 6 to verify certificates.	COMM-004
SEC.099	6.11.10	Certificate verifiers MUST reject certificates that contain one or more unsupported critical extensions.	
SEC.100	6.11.10	Policy-mapping is not supported, and certificates containing policy mappings MUST be rejected.	COMM-004
SEC.101	6.11.10	Name-constraints are not supported and certificates containing name-constraints MUST be rejected.	COMM-004
SEC.102	6.11.10	Any extension not listed by name within this document SHOULD NOT be included within a compliant certificate and, if included, MUST NOT be marked critical.	COMM-004
SEC.103	6.11.10.1	A IEEE 2030.5 device or application MUST NOT use OCSP for the purpose of attempting to validate Manufacturing PKI-issued certificates.	
SEC.104	6.11.11	The site-local multicast address FF05::FB is designated for Extended Multicast DNS and SHALL be used as the destination address for all xmDNS multicast requests and multicast responses.	COMM-001
SEC.105	6.11.11	IEEE 2030.5 SHALL use global addresses or Unique Local Addresses [RFC 4193] in the source address of xmDNS requests and responses.	COMM-001
SEC.106	6.11.11	Guidelines on the use of unicast responses SHALL be employed as described in the xmDNS draft unless noted otherwise.	COMM-001
SEC.107	6.11.11	xmDNS requests from HAN devices that are mains powered SHALL use site-local multicast addressing to ensure that all sub-networks within the HAN are reachable and MAY request unicast responses.	COMM-001
SEC.108	6.11.11	EndDevice servers where the particular HAN device is registered) SHALL employ site-local multicast destination addresses and SHALL request unicast responses.	COMM-001
SEC.109	6.11.11	When a server receives a request with the QU bit set it SHALL return a unicast response.	COMM-001
TIME.001	9.1.3	All devices SHALL implement the Time function set as a resource server, or a client, or both.	CORE-005
TIME.002	9.1.3	All communication of time used throughout the specification, with the exception of time for user display, SHALL be according to the definition of TimeType.	CORE-005
TIME.003	9.1.3	Devices with user displays SHALL support local time and daylight savings time offsets.	CORE-005
TIME.004	9.1.3	If FunctionSetAssignments contain both Event-based function sets (e.g., DRLC, pricing, message) and a Time resource then devices SHALL use the Time resource from the same FunctionSetAssignments when executing the events from the associated Event-based function set.	

Req. ID	Section	Description	Test IDs
TIME.005	9.1.3	When the device executes these events, the device SHALL use the Time resource from the server device from which it received the event.	
TIME.006	9.1.3	If a client discovers an Event-based function set and cannot also retrieve a Time resource from this same server, it SHALL NOT act on the events from this Event-based function set.	
TIME.007	9.1.3	If a device is not processing events it SHALL discover and choose the Time resource with the best quality metric.	CORE-005
TIME.008	9.1.3	If a time server serves an intentionally uncoordinated time in the currentTime field, it SHALL have a quality metric of 7 - time intentionally uncoordinated.	CORE-005, CORE-006
TIME.009	9.1.3	To keep network traffic to a minimum this polling SHALL be no more than once per 15 minutes once a valid time source has been established.	CORE-005, CORE-006
TIME.010	9.1.3	Requests SHALL NOT exceed once per 15 seconds and SHALL employ exponential back off prior to establishing a valid time source.	CORE-005, CORE-006
TIME.011	9.1.3	Time adjustments less than 60 seconds SHALL never be made backwards (e.g., use stall time or long seconds to correct for being ahead on time).	CORE-005, CORE-006
TIME.012	B.1.7	tzOffset attribute (TimeOffsetType) Local time zone offset from currentTime. Does not include any daylight savings time offsets. For American time zones, a negative tzOffset SHALL be used (eg, EST = GMT-5 which is -18000).	CORE-005, CORE-006

Requirements Tested

Test ID	Requirements Tested
Communication Fundamentals Tests	
COMM-001	DNS.001 DNS.002 DNS.003 DNS.004 DNS.005 DNS.006 DNS.007 DNS.008 DNS.009 DNS.010 DNS.011 DNS.012 DNS.013 DNS.014 DNS.015 DNS.016 DNS.017 DNS.018 DNS.019 DNS.020 DNS.021 DNS.022 DNS.023 DNS.024 DNS.025 DNS.026 DNS.027 DNS.028 DNS.029 DNS.030 DNS.031 DNS.032 DNS.033 DNS.034 DNS.036 DNS.037 DNS.038 DNS.039 DNS.040 DNS.041 SEC.104 SEC.105 SEC.106 SEC.107 SEC.108 SEC.109
COMM-002	DNS.003 DNS.004 DNS.040 DNS.041 SEC.029
COMM-003	SEC.015 SEC.016 SEC.017 SEC.020 SEC.021 SEC.022 SEC.023 SEC.024 SEC.027
COMM-004	SEC.035 SEC.041 SEC.042 SEC.049 SEC.050 SEC.051 SEC.056 SEC.057 SEC.058 SEC.059 SEC.060 SEC.061 SEC.064 SEC.069 SEC.098 SEC.100 SEC.101 SEC.102
COMM-005	SEC.013 SEC.025 SEC.026 SEC.027 SEC.031
COMM-006	BASE.009 BASE.076 MULTI.001 SEC.001 SEC.002 SEC.003 SEC.005 SEC.006 SEC.007 SEC.007 SEC.008 SEC.028 SEC.030 SEC.032
Core Function Set Tests	
CORE-001	GEN.039 GEN.042 GEN.043 GEN.045 GEN.047 GEN.048 GEN.052 GEN.053 GEN.054 GEN.056 GEN.058 GEN.059
CORE-002	GEN.037 GEN.038 GEN.040 GEN.041 GEN.043 GEN.044 GEN.045 GEN.047 GEN.048 GEN.049
CORE-003	BASE.009 EVENT.001 EVENT.002
CORE-004	BASE.002 GEN.014 GEN.015 GEN.016 GEN.020 GEN.021 GEN.022 GEN.023 GEN.024 GEN.025 GEN.026 GEN.027 GEN.028 GEN.030 GEN.031 GEN.032 GEN.033 GEN.034 GEN.036
CORE-005	BASE.007 TIME.001 TIME.002 TIME.003 TIME.007 TIME.008 TIME.009 TIME.010
CORE-006	BASE.007 TIME.008 TIME.009 TIME.010 TIME.011 TIME.012
CORE-007	BASE.001
CORE-008	BASE.009 BASE.074 BASE.076 SEC.028 SEC.030 SEC.032
CORE-009	BASE.009 BASE.076 MULTI.001 SEC.001 SEC.002 SEC.003 SEC.005 SEC.006 SEC.007 SEC.007 SEC.008 SEC.028 SEC.030 SEC.032
CORE-010	BASE.004 BASE.005 BASE.006 BASE.007 BASE.008 BASE.009
CORE-011	BASE.074 BASE.075 BASE.076
CORE-012	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.034 DER.041
CORE-013	DER.001 DER.002 DER.003 DER.004 DER.005 DER.006 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.034
CORE-014	DER.016 DER.026 DER.027 DER.028 DER.029 DER.030 DER.034 DER.038 DER.039
CORE-015	DER.009 DER.006 DER.017 DER.026 DER.027 DER.028 DER.029

Test ID	Requirements Tested
	DER.030 DER.034 DER.038 DER.039
CORE-016	DER.009 DER.006 DER.017 DER.026 DER.027 DER.028 DER.029 DER.030 DER.034 DER.038 DER.039
CORE-017	DER.009 DER.006 DER.017 DER.026 DER.027 DER.028 DER.029 DER.030 DER.034 DER.038 DER.039
CORE-018	BASE.013 BASE.014 BASE.021 BASE.022 BASE.023 BASE.024 BASE.026 BASE.028 BASE.029 BASE.063 BASE.064 BASE.065 BASE.066 BASE.067 BASE.068 BASE.069
CORE-019	BASE.024 BASE.025 BASE.026 BASE.027 BASE.028 BASE.029 BASE.063 BASE.064 BASE.065 BASE.066 BASE.067 BASE.068 BASE.069 BASE.070
CORE-020	MUP.001 MUP.002 MUP.003 MUP.004 MUP.005 MUP.006 MUP.007 MUP.008 MUP.009 MUP.010 MUP.011 MUP.012 MUP.013 MUP.014 MUP.015 MUP.016 MUP.017 MUP.018 MUP.019
CORE-021	EVENT.011 EVENT.013 EVENT.014 EVENT.021 EVENT.022 EVENT.024 RAND.001 RAND.002 RAND.004 RAND.005 RAND.006 RAND.008 RAND.009 RAND.010 RAND.012 RAND.013 RAND.014 RAND.015 RAND.016 RAND.017
CORE-022	BASE.030 BASE.031 BASE.032 BASE.033 BASE.034 BASE.033 BASE.072
Basic Functions Tests	
BASIC-001	BASE.074 SEC.001 SEC.002 SEC.004 SEC.005
BASIC-002	DER.001 DER.002 DER.003 DER.004 DER.005 DER.006 DER.007 DER.008 DER.009 DER.010 DER.011 DER.012
BASIC-003	DER.001 DER.002 DER.003 DER.004 DER.005 DER.006 DER.007 DER.008 DER.009 DER.010 DER.011 DER.012
BASIC-004	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038
BASIC-005	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038
BASIC-006	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038
BASIC-007	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038
BASIC-008	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038
BASIC-009	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038
BASIC-010	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038
BASIC-011	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038
BASIC-012	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038
BASIC-013	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038
BASIC-014	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008

Test ID	Requirements Tested
	DER.010 DER.011 DER.012 DER.013 DER.038
BASIC-015	DER.001 DER.002 DER.003 DER.004 DER.005 DER.006 DER.008 DER.009 DER.010 DER.011 DER.012 DER.013 DER.042 EVENT.001 EVENT.002 EVENT.004 EVENT.005 EVENT.009 EVENT.021 EVENT.022 EVENT.026 EVENT.031 EVENT.032 EVENT.033 EVENT.034 EVENT.042
BASIC-016	DER.001 DER.002 DER.003 DER.005 DER.007 DER.008 DER.009
BASIC-017	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.034 EVENT.042
BASIC-018	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.034 EVENT.042
BASIC-019	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.034 EVENT.042
BASIC-020	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.034 EVENT.042
BASIC-021	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.017 EVENT.019 EVENT.034 EVENT.042
BASIC-022	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.034 EVENT.042
BASIC-023	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.034 EVENT.042
BASIC-024	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.028 EVENT.034 EVENT.042
BASIC-025	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.028 EVENT.034 EVENT.042
BASIC-026	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.028 EVENT.034 EVENT.042
BASIC-027	LOG.001 LOG.002 LOG.003 LOG.004 LOG.005 LOG.006 LOG.008
BASIC-028	BASE.009 BASE.045 BASE.074 BASE.076 DER.009 DER.006 DER.007 DER.026 DER.027 DER.028 DER.029 DER.030 DER.034 DER.038 DER.039 SEC.030
BASIC-029	MUP.001 MUP.002 MUP.003 MUP.004 MUP.005 MUP.006 MUP.007 MUP.008 MUP.009 MUP.010 MUP.011 MUP.012 MUP.013 MUP.014 MUP.015 MUP.016 MUP.017 MUP.018 MUP.019
Utility Server Aggregator Model Tests	

Test ID	Requirements Tested
UTIL-001	BASE.074 SEC.001 SEC.002 SEC.004 SEC.005
UTIL-002	BASE.009 BASE.045 BASE.074 BASE.076 DER.009 DER.006 DER.007 DER.026 DER.027 DER.028 DER.029 DER.030 DER.034 DER.038 DER.039 SEC.030
UTIL-003	BASE.004 BASE.005 BASE.006 BASE.007 BASE.008 BASE.009
UTIL-004	BASE.002 BASE.004 BASE.005 BASE.006 BASE.007 BASE.008 BASE.009 BASE.045 BASE.074 BASE.076 DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.001 EVENT.034 EVENT.042
Aggregator Operation Tests	
AGG-001	BASE.013 BASE.014 BASE.022 BASE.023 BASE.024 BASE.026 BASE.029 BASE.063 BASE.064 BASE.065 BASE.066 BASE.067 BASE.068 BASE.069
AGG-002	DER.001 DER.002 DER.003 DER.005 DER.007 DER.008 DER.009
AGG-003	DER.001 DER.002 DER.003 DER.004 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.034 EVENT.042
AGG-004	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.034 EVENT.042
AGG-005	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.034 EVENT.042
AGG-006	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.034 EVENT.042
AGG-007	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.008 EVENT.09 EVENT.034 EVENT.042
AGG-008	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.008 EVENT.0017 EVENT.034 EVENT.042
AGG-009	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.008 EVENT.009 EVENT.018 EVENT.019 EVENT.034 EVENT.042
AGG-010	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.028 EVENT.034 EVENT.042
AGG-011	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.028 EVENT.034 EVENT.042
AGG-012	DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.009 EVENT.001 EVENT.002 EVENT.003 EVENT.004 EVENT.005 EVENT.009 EVENT.028 EVENT.034 EVENT.042

Test ID	Requirements Tested
Error Handling Tests	
ERR-001	GEN.037 GEN.038 GEN.039 GEN.042 DNS.042
ERR-002	BASE.024 BASE.025 BASE.026 BASE.027 BASE.028 BASE.029 BASE.063 BASE.064 BASE.065 BASE.066 BASE.067 BASE.068 BASE.069 BASE.070
Maintenance of the Model Tests	
MAINT-001	BASE.024 BASE.025 BASE.026 BASE.027 BASE.028 BASE.029 BASE.063 BASE.064 BASE.065 BASE.066 BASE.067 BASE.068 BASE.069 BASE.070
MAINT-002	BASE.024 BASE.025 BASE.026 BASE.027 BASE.028 BASE.029 BASE.063 BASE.064 BASE.065 BASE.066 BASE.067 BASE.068 BASE.069 BASE.070
MAINT-003	BASE.024 BASE.025 BASE.026 BASE.027 BASE.028 BASE.029 BASE.063 BASE.064 BASE.065 BASE.066 BASE.067 BASE.068 BASE.069 BASE.070 DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038
MAINT-004	BASE.024 BASE.025 BASE.026 BASE.027 BASE.028 BASE.029 BASE.063 BASE.064 BASE.065 BASE.066 BASE.067 BASE.068 BASE.069 BASE.070 DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038
MAINT-005	BASE.024 BASE.025 BASE.026 BASE.027 BASE.028 BASE.029 BASE.063 BASE.064 BASE.065 BASE.066 BASE.067 BASE.068 BASE.069 BASE.070 DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038
MAINT-006	BASE.024 BASE.025 BASE.026 BASE.027 BASE.028 BASE.029 BASE.063 BASE.064 BASE.065 BASE.066 BASE.067 BASE.068 BASE.069 BASE.070 DER.001 DER.002 DER.003 DER.004 DER.005 DER.007 DER.008 DER.010 DER.011 DER.012 DER.013 DER.038