BACnet® TESTING LABORATORIES ADDENDA

Addendum imp1 to BTL Test Package 26.0

Revision final
Revised 9/9/2025

Approved by the BTL Working Group on June 19, 2025;
Approved by the BTL Working Group Voting Members on September 23, 2025;
Published on September 24, 2025.

[This foreword and the "Overview" on the following pages are not part of this Test Package. They are merely informative and do not contain requirements necessary for conformance to the Test Package.]

FOREWORD

The purpose of this addendum is to present current changes being made to the BTL Test Package. These modifications are the result of change proposals made pursuant to the continuous maintenance procedures and of deliberations within the BTL-WG Committee. The changes are summarized below.

In the following document, language to be added to existing clauses within the BTL Test Package 26.0 is indicated through the use of *italics*, while deletions are indicated by ~~strikethrough~~. Where entirely new subclauses are proposed to be added, plain type is used throughout.

In contrast, changes to BTL Specified Tests also contain a <mark>yellow</mark> highlight to indicate the changes made by this addendum. When this addendum is applied, all highlighting will be removed. Change markings on tests will remain to indicate the difference between the new test and an existing 135.1 test. If a test being modified has never existed in 135.1, the applied result should not contain any change markings. When this is the case, square brackets will be used to describe the changes required for this test.

Each addendum can stand independently unless specifically noted via dependency within the addendum. If multiple addenda change the same test or section, each future released addendum that changes the same test or section will note in square brackets whether or not those changes are reflected.

BTL-26.0 imp1-1: Add NPO for Virtual Datalink [BTLWG-1683]

Overview:

Network_Type = VIRTUAL for a NPO is not included in the Test Package. This NPO is required for all devices that claim GW-VN-B as of PR 17.

From 12.56.9. "VIRTUAL - Indicates that this port represents the configuration and properties of a virtual network as described in Clauses H.1.1.1 and H.1.1.2."

Changes:

Checklist Changes

9 Data Link Layer

...			
Data Link Layer - Virtual Network			
	R	Base Requirements	
	R	Supports GW-VN-B	
	C[1]	Supports the Network Port object	
	C[1,2]	Supports writable Network_Number property	
	O[1]	Supports configurable Out_Of_Service property	
	C[1,3]	Supports hierarchical Network Port objects	
	C[1,3]	Supports Non-hierarchical Network Port objects	
	O[1]	Supports the Routing_Table property	
	O[1]	Supports the Network Port object Command property	
	O[1,4]	Supports the DISCARD_CHANGES command	
	O[1,4]	Supports the RENEW_FD_REGISTRATION command	
	O[1,4]	Supports the RESTART_SLAVE_DISCOVERY command	
	O[1,4]	Supports the RENEW_DHCP command	
	O[1,4]	Supports the RESTART_AUTONEGOTIATION command	
	O[1,4]	Supports the RESTART_PORT command	
	O[4,5]	Supports the GENERATE_CSR_FILE command	
	O[4,5]	Supports the VALIDATE_CHANGES command	
	[1] Protocol_Revision 17 or higher must be claimed. [2] The Network_Number property shall be present and writable. [3] At least one of these options is required. [4] At least one of these options is required if the Network Port object Command property is supported. [5] Protocol_Revision 24 or higher must be claimed.		
...			

11 Gateway

Support	Listing	Option
Gateway - Virtual Network - B		

Support	Listing	Option
	R	Base Requirements
	C[1]	Supports Data Link Layer - Virtual Network
		[1] Required if the device claims Protocol Revision 17 or higher.
Gateway - Embedded Objects - B		
	R	Base Requirements
	O	Supports command prioritization

Test Plan Changes

[Change Clause 9.10.1]

9.10.1 Base Requirements

There are no base requirements for this section.

[Add Clauses 9.10.2 to 9.10.17]

9.10.2 Supports GW-VN-B

The IUT supports GW-VN-B.

Verify Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims support for GW-VN-B.
	Testing Hints	

9.10.3 Supports the Network Port Object

The IUT contains a Network Port object with Network Type = VIRTUAL.

Verify Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims support for DS-WP-B.
	Testing Hints	
135.1-2023 - 7.3.2.46.1.1 - Configure Network Through Network Port Object Test		
	Test Conditionality	Must be executed.
	Test Directives	Execute this test at least once on each Network Port object that has Network_Type = VIRTUAL and contains writable properties. Execute test 7.3.2.46.1.1 at least once with ReinitializeDevice WARMSTART and at least once with ACTIVATE_CHANGES.
	Testing Hints	
BTL - 7.3.2.46.1.3 - Network Port Non-Volatility Properties Test		
	Test Conditionality	Must be executed if any writable properties are supported for which the values are required for proper operation of the network.
	Test Directives	
	Testing Hints	
135.1-2023 - 7.3.2.46.1.4 - Network Port Configuration Conflict Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	

9.10.4 Supports Writable Network_Number Property

The IUT contains a Network Port object with Network Type = VIRTUAL and Protocol_Level = BACNET_APPLICATION that contains a writable Network_Number property.

135.1-2023 - 7.3.2.46.2 - Network-Number-Is Updates Network_Number_Quality Test		
	Test Conditionality	For IUTs which do not accept a value of zero in their Network_Number property, this test shall be skipped.
	Test Directives	
	Testing Hints	

9.10.5 Supports Configurable Out_Of_Service Property

The IUT contains a Network Port object with Network Type = VIRTUAL and contains a writable or configurable Out_Of_Service property.

BTL - 7.3.1.1.X5 - Out_Of_Service, Status_Flags, Reliability and Command Property Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	

9.10.6 Supports Hierarchical Network Port Objects

The IUT contains a Network Port object with Network Type = VIRTUAL, Protocol_Level = BACNET_APPLICATION and supports a set of Network Port objects which form a hierarchy of Network Port objects.

Verify EPICS		
	Test Conditionality	Must be executed if the IUT claims Protocol_Revision >= 24.
	Test Directives	Verify that each hierarchical Network Port object contains only required and optional properties based on its Network_Type and Protocol_Level.
	Testing Hints	
BTL - 7.3.2.46.X.Y - Valid Hierarchy Chain - Virtual Test		
	Test Conditionality	Must be executed.
	Test Directives	Repeat this test on all Network Port objects with Protocol_Level = BACNET_APPLICATION, Network_Type = VIRTUAL, and Reference_Port <> 4194303.
	Testing Hints	
BTL - 7.3.2.46.4.2 - Properties in Referenced Network Port Reflected in Top Network Port Object		
	Test Conditionality	Must be executed if the IUT claims Protocol Revision < 24.
	Test Directives	
	Testing Hints	
135.1-2023 - 7.3.2.46.4.3 - Changes Reflected in Top Network Port Object		
	Test Conditionality	Must be executed if the IUT claims Protocol Revision < 24 and supports writable properties in its Network Port hierarchies.
	Test Directives	
	Testing Hints	
135.1-2023 - 7.3.2.46.4.4 - Changes Reflected in Lower Network Port Objects		
	Test Conditionality	Must be executed if the IUT claims Protocol Revision < 24 and supports writable properties in its Network Port hierarchies.
	Test Directives	
	Testing Hints	

9.10.7 Supports Non-hierarchical Network Port Objects

The IUT contains a Network Port object with Network Type = VIRTUAL, Protocol_Level = BACNET_APPLICATION and supports non-hierarchical Network Port objects.

Verify EPICS		
	Test Conditionality	Must be executed.

	Test Directives	Verify IUT contains non-hierarchical Network Port objects with Protocol_Level equal to BACNET_APPLICATION for this Network_Type.
	Testing Hints	
Verify EPICS		
	Test Conditionality	Must be executed if the IUT claims Protocol_Revision < 24.
	Test Directives	Verify the Reference_Port is absent or equal to 4194303 for all non-hierarchical Network Port objects of this Network_Type.
	Testing Hints	
Verify EPICS		
	Test Conditionality	Must be executed if the IUT claims Protocol_Revision >= 24.
	Test Directives	Verify the Reference_Port is absent for all non-hierarchical Network Port objects of this Network_Type.
	Testing Hints	
Verify EPICS		
	Test Conditionality	Must be executed.
	Test Directives	Verify the Additional_Reference_Ports property is absent for all non-hierarchical Network Port objects of this Network_Type.
	Testing Hints	
Verify EPICS		
	Test Conditionality	Must be executed if the IUT claims Protocol_Revision >= 24.
	Test Directives	Verify each non-hierarchical Network Port object contains all required properties for this Network_Type.
	Testing Hints	
Verify EPICS		
	Test Conditionality	Must be executed if the IUT claims Protocol_Revision >= 24.
	Test Directives	Verify each non-hierarchical Network Port object contains only valid optional properties for this Network_Type.
	Testing Hints	

9.10.8 Supports the Routing_Table Property

The IUT contains a Network Port object with Network Type = VIRTUAL and Protocol_Level = BACNET_APPLICATION that contains the Routing_Table property.

135.1-2023 - 7.3.2.46.6 - Routing_Table Test		
	Test Conditionality	If the IUT only supports 1 entry in its routing table, then this test shall be skipped.
	Test Directives	
	Testing Hints	

9.10.9 Supports the Network Port Object Command Property

The IUT contains a Network Port object with Network Type = VIRTUAL and supports the Command property.

135.1-2023 - 7.3.2.46.3.1 - IDLE Command Rejected		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 7.3.2.46.3.9 - No Commands if Changes_Pending Test		
	Test Conditionality	Must be executed if the Network Port object supports DISCARD_CHANGES and at least 1 other non-IDLE command.
	Test Directives	
	Testing Hints	
BTL - 7.3.2.46.3.2.X2 - DISCARD_CHANGES Command Failure Test		
	Test Conditionality	Must be executed if the Network Port object does not support the DISCARD_CHANGES command.
	Test Directives	
	Testing Hints	

135.1-2023 - 7.3.2.46.3.3.2 - RENEW_FD_REGISTRATION Command Failure Test		
	Test Conditionality	Must be executed if the Network Port object does not support the RENEW_FD_REGISTRATION command.
	Test Directives	
	Testing Hints	
135.1-2023 - 7.3.2.46.3.4.2 - RESTART_SLAVE_DISCOVERY Command Failure Test		
	Test Conditionality	Must be executed if the Network Port object does not support the RESTART_SLAVE_DISCOVERY command.
	Test Directives	
	Testing Hints	
135.1-2023 - 7.3.2.46.3.5.2 - RENEW_DHCP Command Failure Test		
	Test Conditionality	Must be executed if the Network Port object does not support the RENEW_DHCP command.
	Test Directives	
	Testing Hints	
135.1-2023 - 7.3.2.46.3.6.2 - RESTART_AUTONEGOTIATION Command Failure Test		
	Test Conditionality	Must be executed if the Network Port object does not support the RESTART_AUTONEGOTIATION command.
	Test Directives	
	Testing Hints	
135.1-2023 - 7.3.2.46.3.7.2 - DISCONNECT Command Failure Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
135.1-2023 - 7.3.2.46.3.8.2 - RESTART_PORT Command Failure Test		
	Test Conditionality	Must be executed if the Network Port object does not support the RESTART_PORT command.
	Test Directives	
	Testing Hints	
BTL - 7.3.2.46.3.X.2 - GENERATE_CSR_FILE Command Failure Test		
	Test Conditionality	Must be executed if the Network Port object does not support the GENERATE_CSR_FILE command.
	Test Directives	
	Testing Hints	
BTL - 7.3.2.46.3.X.4 - VALIDATE_CHANGES Command Failure Test		
	Test Conditionality	Must be executed if the Network Port object does not support the VALIDATE_CHANGES command.
	Test Directives	
	Testing Hints	

9.10.10 Supports the DISCARD_CHANGES Command

The IUT contains a Network Port object with Network Type = VIRTUAL and supports the DISCARD_CHANGES command.

BTL - 7.3.2.46.3.2.X1 - DISCARD_CHANGES Command Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	

9.10.11 Supports the RENEW_FD_REGISTRATION Command

The IUT contains a Network Port object with Network Type = VIRTUAL, Protocol_Level = BACNET_APPLICATION, and supports the RENEW_FD_REGISTRATION command.

135.1-2023 - 7.3.2.46.3.3.1 - RENEW_FD_REGISTRATION Command Test		
	Test Conditionality	Must be executed.
	Test Directives	

Testing Hints	

9.10.12 Supports the RESTART_SLAVE_DISCOVERY Command

The IUT contains a Network Port object with Network Type = VIRTUAL and Protocol_Level = BACNET_APPLICATION and supports the RESTART_SLAVE_DISCOVERY command.

135.1-2023 - 7.3.2.46.3.4.1 - RESTART_SLAVE_DISCOVERY Command Test	
Test Conditionality	Must be executed.
Test Directives	
Testing Hints	

9.10.13 Supports the RENEW_DHCP Command

The IUT contains a Network Port object with Network Type = VIRTUAL, Protocol_Level = BACNET_APPLICATION or PROTOCOL, and supports the RENEW_DHCP command.

135.1-2023 - 7.3.2.46.3.5.1 - RENEW_DHCP Command Test	
Test Conditionality	Must be executed.
Test Directives	
Testing Hints	

9.10.14 Supports the RESTART_AUTONEGOTIATION Command

The IUT contains a Network Port object with Network Type = VIRTUAL, Protocol_Level = BACNET_APPLICATION or PHYSICAL and supports the RESTART_AUTONEGOTIATION command.

135.1-2023 - 7.3.2.46.3.6.1 - RESTART_AUTONEGOTIATION Command Test	
Test Conditionality	Must be executed.
Test Directives	
Testing Hints	

9.10.15 Supports the RESTART_PORT Command

The IUT contains a Network Port object with Network Type = VIRTUAL and supports the RESTART_PORT command.

135.1-2023 - 7.3.2.46.3.8.1 - RESTART_PORT Command Test	
Test Conditionality	Must be executed.
Test Directives	
Testing Hints	

9.10.16 Supports the GENERATE_CSR_FILE Command

The IUT contains a Network Port object with Network Type = VIRTUAL, Protocol_Level = BACNET_APPLICATION, and supports the GENERATE_CSR_FILE Command.

BTL - 7.3.2.46.3.X.1 - GENERATE_CSR_FILE Command Test	
Test Conditionality	Must be executed if the IUT supports the GENERATE_CSR_FILE command.
Test Directives	
Testing Hints	

9.10.17 Supports the VALIDATE_CHANGES Command

The IUT contains a Network Port object with Network Type = VIRTUAL and supports the VALIDATE_CHANGES command.

BTL - 7.3.2.46.3.X.3 - VALIDATE_CHANGES Command Test	
Test Conditionality	Must be executed.
Test Directives	

Testing Hints	

Specified Test Changes

[Add 7.3.2.46.X.Y]

7.3.2.46.X Valid Hierarchy Tests

7.3.2.46.X.Y Valid Hierarchy Chain - Virtual Test

Reason for Change: Explicit test for this data link.

Purpose: To verify that a hierarchical chain of BACnet network port objects in the IUT is organized in a valid hierarchy of protocol levels.

Test Concept: Starting with a hierarchical Network Port object (NP_APP) and Network_Type which represents a configured application layer port (BACNET_APPLICATION), verify this port contains valid Network_Type, Protocol_Level, and Reference_Port. Visit each Network Port object in the hierarchy ensuring that the Network_Type, Protocol_Level, and Reference_Port are valid. Using the Additional_Reference_Ports property, visit each Network Port object in the hierarchy ensuring that the Network_Type, Protocol_Level, Reference_Port, and Additional_Reference_Ports are valid.

Notes to Tester: NPOs of this Network_Type are excluded from this test if they are at Protocol_Level = PROTOCOL and they are not referenced by another NPO. NPOs of this Network_Type are also excluded from this test if they are at Protocol_Level = PHYSICAL and they are not referenced by another NPO.

Test Steps:

```
-- Verify Protocol_Level = BACNET_APPLICATION
1.    VERIFY NP_APP, Protocol_Level = BACNET_APPLICATION
2.    VERIFY NP_APP, Network_Type = VIRTUAL
3.    RP = READ NP_APP, Reference_Port
4.    VERIFY RP <> 4194303

-- Check the rest of the chain
5.    WHILE (RP <> 4194303)
      {
6.        NPx = (Network Port, RP)
7.        VERIFY NPx, Protocol_Level <> (NON_BACNET_APPLICATION OR BACNET_APPLICATION)
          -- check that if PHYSICAL level it is the last NPO
8.        IF (NPx, Protocol_Level == PHYSICAL) THEN
9.            VERIFY NPx, Reference_Port = 4194303
10.       RP = READ NPx, Reference_Port
      }
      -- either the NPO is at PROTOCOL and does not reference another NPO or it is PHYSICAL

11.   IF (Protocol_Revision >= 24 and NP_APP, Additional_Reference_Ports is present) THEN {
12.       REPEAT ARP = (for each entry in NP_APP, Additional_Reference_Ports) DO {
13.           NPx = (Network Port, ARP)
14.           VERIFY NPx, Protocol_Level <> (NON_BACNET_APPLICATION or BACNET_APPLICATION)
              -- check that if PHYSICAL level it is the last NPO
15.           IF (NPx, Protocol_Level == PHYSICAL) THEN
16.               IF (NPx, Additional_Reference_Ports is present) THEN
17.                   VERIFY NPx, Additional_Reference_Ports = (empty list)
              -- Check the rest of the chain
18.           RP = READ NPx, Reference_Port
19.           WHILE (RP <> 4194303)
                  {
```

```
20.              NPx = (Network Port, RP)
21.              VERIFY NPx, Protocol_Level <> (NON_BACNET_APPLICATION
                        or BACNET_APPLICATION)
                 -- check that if PHYSICAL level it is the last NPO
22.              IF (NPx, Protocol_Level == PHYSICAL) THEN
23.                     VERIFY NPx, Reference_Port = 4194303
24.              RP = READ NPx, Reference_Port
            }
        }
    }
```

BTL-26.0 imp1-2: Test 9.39.2 Should not use Internal Processing Fail Time [BTLWG-133]

Overview:

Using Internal Processing Fail Time to see if the IUT resets does not make any sense. Why does the test care if the IUT resets?

Changes:

Checklist Changes

Test Plan Changes

2 Basic Functionality (Applies To All BACnet Devices)

2.1.1 Base Requirements

All BACnet devices must meet these base requirements.

...		
~~135.1 2023~~**BTL** - 9.39.2 - Unsupported Unconfirmed Services Test		
	Test Conditionality	Must be executed.
	Test Directives	Don't forget Unconfirmed Private Transfer; if the IUT supports Unconfirmed Private Transfer, send one with a different vendor Id. Also try services that have not yet been defined (reserved enumeration range), sending them in a BACnet-Unconfirmed-Request-PDU with DER bit clear and with no InvokeID.
	Testing Hints	
...		

Specified Test Changes

[Move test 9.39.2 from 135.1-2023 into BTL Specified Tests, and modified]

9.39.2   Unsupported Unconfirmed Services Test
Purpose: This test case verifies that the IUT will quietly accept and discard any unconfirmed services that it does not support. When determining the set of services to send to the IUT, the UnconfirmedPrivateTransfer service should be included regardless of whether the IUT supports it or not. The UnconfirmedPrivateTransfer service shall be sent with a vendor ID/Service Number pair not supported by the device.

Configuration Requirements: This test requires that the IUT be placed into a normal operating state ~~in which it will not initiate any requests~~.

Test Steps:

1. READ S1 = System_Status
2. VERIFY S1 = ~~System_Status ==~~ OPERATIONAL | OPERATIONAL_READ_ONLY
3. REPEAT X = (all unconfirmed services that the IUT does not execute) DO {
       TRANSMIT X
       WHILE (Unconfirmed Response Fail Time) DO {
           CHECK (the IUT does not send any packets in response to X)
       }

       VERIFY System_Status = S1
       BEFORE **Internal Processing Fail Time**
           CHECK (verify that the IUT did not reset and that the IUT did not send any packets)
       VERIFY System_Status = (the value of System_Status read in step 1)
  }

Passing Result: The IUT does not reset and sends no packets in response to the services.

BTL-26.0 imp1-3: Add Negative Test for Property Array Index [BTLWG-0455]

Overview:

It is needed to add negative test of 'Property Array Index' = 0 for PROPERTY_IS_NOT_AN_ARRAY.
In current 9.22.2.1, it has only testing 'Property Array Index' = (any positive integer) for PROPERTY_IS_NOT_AN_ARRAY.

Changes:

Checklist Changes

None

Test Plan Changes

[In BTL Test Plan, change all occurrences of test 9.22.2.1 from 135.1 to BTL]

Specified Test Changes

[Move test 9.22.2.1 from 135.1 to BTL]

9.22.2.1 Writing Non-Array Properties with an Array Index
Reason for Change: Add negative testing when index = 0.

Purpose: To verify that the IUT can execute WriteProperty service requests when the property value is not an array but an array index is included in the service request.

Test Concept: The TD shall select an object in the IUT that contains a writable scalar property designated P1. An attempt will be made to write to this property using an ARRAY INDEX *that is a positive integer and again with and an ARRAY INDEX = 0*. If no suitable object can be found, then this test shall be omitted.

Configuration Requirements: If the IUT supports any writable properties that are scalars, it shall be configured with at least one such property that can be used for this test.

Test Steps:
1. READ X = (Object1), P1
2. TRANSMIT WriteProperty-Request,
    'Object Identifier' = Object1,
    'Property Identifier' = P1,
    'Property Value' = (any valid value of the correct datatype for this property subject to
            the restrictions specified in the EPICS as defined in 4.4.2, except
            the value X read in step 1),
    'Property Array Index' = (any positive integer)
3. IF (Protocol_Revision is present AND Protocol_Revision >= 4) THEN
        RECEIVE BACnet-Error PDU,
            Error Class = PROPERTY,
            Error Code = PROPERTY_IS_NOT_AN_ARRAY
  ELSE
        RECEIVE BACnet-Error PDU,
            Error Class = SERVICES,
            Error Code = INCONSISTENT_PARAMETERS
4. VERIFY (Object1), P1 = X
5. TRANSMIT WriteProperty-Request,
    'Object Identifier' = Object1,

BTL-26.0 imp1-4: Test 9.18.1.6 Incorrectly Suggests the Use of Read Range [BTLWG-1386]

Overview:

Test 9.18.1.6 (9.18.1.X3) checks that the IUT abides the max-segments-accepted request header value when building a response. The test expects an Abort-PDU when the response would exceed the indicated response size accepted.

In the notes to tester, the test suggests ReadRange as one of the services that could be used, but a proper implementation of ReadRange would not result in an abort as it would send fewer results with the More Items flag set.

At a minimum, the notes to tester should be changed as follows:
Hints to Tester: An attempt to read the whole Object_List might suffice. Or a ReadRange or ReadPropertyMultiple or AtomicReadFile request, if any of those services are executed.

Changes:

Checklist Changes

None

Test Plan Changes

[Change all occurrences of test 9.18.1.6 from 135.1 to BTL]

2.2 Segmentation Support

2.2.2   Issues Segmented Responses

BACnet devices which are capable of issuing segmented responses shall meet these requirements.

~~135.1-2023~~*BTL* - 9.18.1.6 - Respects max-segments-accepted bit pattern		
	Test Conditionality	Must be executed, if there is any response generated by IUT which is large enough to require segmentation. *If the largest response the IUT can return is 100 octets or less, then this test shall be skipped.*
	Test Directives	Test with a 'max-segments-accepted' bit pattern *whose* value ~~that~~ is less than ~~TD's Max_Segments_Accepted property value, to ascertain that the 'max-segments-accepted' bit pattern value is what governs.~~*the value of the TD's Max_Segments_Accepted property.*
	Testing Hints	

Specified Test Changes

9.18.1.6 Respects max-segments-accepted bit pattern

Reason for Change: Update test to include missing parameter in Step 1. Remove ReadRange as one of the suggested services that could be used for the test, since a proper implementation of ReadRange will not result in an abort and would instead send fewer results with the More Items flag set.

Purpose: To verify that the IUT abides by the 'max-segments-accepted' parameter, when the size of the response ~~does require~~*requires* segmentation.

Configuration Requirements: ~~Use a very small 50 octet 'max-APDU-length-accepted' size in the request. The BACnetConfirmed-Request-PDU shall be one where the response size will exceed 2 times 'max-APDU-length-accepted' and so require at least three segments. If the largest response that the IUT can return is 100 or fewer octets, then this test shall be skipped.~~ *The TD shall be configured to issue a BACnet-Confirmed-Request-PDU, specifying a max-apdu-length-accepted of 50-octets, max-segments-accepted equal to 2, and segmented-response-accepted equal to True. The total response from the IUT should exceed three 50-octet segments, triggering the IUT to respond with a BUFFER_OVERFLOW.*

Notes to Tester: ~~An attempt to read the whole Object_List might suffice. Or a ReadRange or ReadPropertyMultiple or AtomicReadFile request, if any of those services are executed.~~*An attempt to read the whole Object List may satisfy the requirements of this test. If the IUT supports execution of ReadPropertyMultiple service or AtomicReadFile service, then one of those services may also satisfy the requirements of this test.*

Test Steps:

1. TRANSMIT BACnet-Confirmed-Request-PDU,
         'segmented-response-accepted' = TRUE
         'max-segments-accepted' = ~~2~~*'B'001', -- 2 segments accepted*
         'max-apdu-length-accepted' = B'0000', -- minimum message size, 50 octets
2. RECEIVE BACnet-Abort-PDU,
         'Abort Reason' = BUFFER_OVERFLOW

BTL-26.0 imp1-5: Clarified the DESTINATION in Several Tests [BTLWG-1316]

Overview:

Execution of Who-Is Service Requests Originating from a Remote Network define Destination for received I-Am-Request as:

DESTINATION = GLOBAL BROADCAST | REMOTE BROADCAST (to the network specified by SNET in step 1) | TD

Problem is the unicast "TD". The original WhoIs in step 1 did not come from the TD but the TD "faked" the request to pretend it came from some device in another network. So if the IUT sends an unicast Iam it should not go to the TD but it should go to the faked device from step 1.

Tests 9.33.2.1, 9.33.2.2 (both 135.1-2019) and 9.33.2.3 (Specified Tests 20.0.1)

Changes:

Checklist Changes

None

Test Plan Changes

[Move tests 9.33.2.1 and 9.33.2.2 from 135.1 to BTL]

Specified Test Changes

9.33.2.1 General Inquiry, Global Broadcast from a Remote Network

Reason for Change: Clarify the DINFO designations of the expected response.

Purpose: To verify the ability of the IUT to recognize the origin of a globally broadcast Who-Is service request and respond such that the device originating the request receives the response.

Test Steps:

1. TRANSMIT
   DESTINATION = GLOBAL BROADCAST,
   SNET = ~~(any remote network number~~ X, any remote network number~~)~~,
   SADR = ~~(any MAC address valid for the specified network~~ Y, any MAC address valid for the specified network~~)~~,
   Who-Is-Request
2. BEFORE **Unconfirmed Response Fail Time**
   RECEIVE
   ~~DESTINATION =~~ ~~GLOBAL BROADCAST~~
   ~~| REMOTE BROADCAST (to the network specified by SNET in step 1)~~
   ~~| TD ,~~
   DESTINATION = *GLOBAL BROADCAST | REMOTE BROADCAST (to network X)| TD (DNET = X, DADR = Y),*
   I-Am-Request,
   'I Am Device Identifier' = (the IUT's Device object),
   'Max APDU Length Accepted' = (the value specified in the EPICS),
   'Segmentation Supported' = (the value specified in the EPICS),

'Vendor Identifier' =   (the identifier registered for this vendor)

9.33.2.2 General Inquiry, Remote Broadcast

Reason for Change: Clarify the DINFO designations of the expected response.

Purpose: To verify the ability of the IUT to recognize the origin of a remotely broadcast Who-Is service request and respond such that the device originating the request receives the response.

Test Steps:

1. TRANSMIT
        DESTINATION =   LOCAL BROADCAST,
        SNET =   ~~(any remote network number~~ X, any remote network number~~)~~,
        SADR =   ~~(any MAC address valid for the specified network~~ Y, any MAC address
valid for the specified network~~)~~,
        Who-Is-Request
2. BEFORE **Unconfirmed Response Fail Time**
        RECEIVE
        ~~DESTINATION =   GLOBAL BROADCAST~~
        ~~| REMOTE BROADCAST (to the network specified by SNET~~
        ~~in step 1)~~
        ~~| TD ,~~
        DESTINATION =   GLOBAL BROADCAST | REMOTE BROADCAST (to network X)| TD
                                           (DNET = X, DADR = Y),

        I-Am-Request,
        'I Am Device Identifier' =   (the IUT's Device object),
        'Max APDU Length Accepted' =   (the value specified in the EPICS),
        'Segmentation Supported' =   (the value specified in the EPICS),
        'Vendor Identifier' =   (the identifier registered for this vendor)

9.33.2.3 General Inquiry, Directed to a Remote Device

Reason for Change: Clarify the DINFO designations of the expected response.

Purpose: To verify that the IUT responds with an I-Am service that is of the form global broadcast, remote broadcast or unicast.

Test Steps:

1. TRANSMIT
        DESTINATION =   IUT,
        SNET =   ~~(any remote network number~~ X, any remote network number~~)~~,
        SADR =   ~~(any MAC address valid for the specified network~~ Y, any MAC address
valid for the specified network~~)~~,
        Who-Is-Request
2. BEFORE **Unconfirmed Response Fail Time**
        RECEIVE
        ~~DESTINATION =   GLOBAL BROADCAST~~
        ~~| REMOTE BROADCAST (to the network specified by SNET~~
        ~~in step 1)~~
        ~~| TD ,~~
        DESTINATION =   GLOBAL BROADCAST | REMOTE BROADCAST (to network X)| TD
                                           (DNET = X, DADR = Y),

        I-Am-Request,
        'I Am Device Identifier' =   (the IUT's Device object),
        'Max APDU Length Accepted' =   (the value specified in the EPICS),
        'Segmentation Supported' =   (the value specified in the EPICS),
        'Vendor Identifier' =   (the identifier registered for this vendor)

BTL-26.0 imp1-6: Remove Deprecated A-side Alarm BIBBs [BTLWG-1330]

Overview:

AE-ASUM-A, AE-ESUM-A, and AE-INFO-A were replaced with AE-AS-A in 135-2008 Add l (PR 9). Time to remove these BIBBs.

Changes:

Checklist Changes

...		
Alarm and Event Management - Alarm Summary - A		
	R¹	Base Requirements
	¹ This BIBB, AE-ASUM-A, can be claimed if AE-AS-A is also claimed.	
Alarm and Event Management - Alarm Summary - B		
	R	Base Requirements
Alarm and Event Management - Enrollment Summary - A		
	R¹	Base Requirements
	O	Generates requests with an Enrollment filter
	O	Generates requests with an Event State filter
	O	Generates requests with an Event Type filter
	O	Generates requests with a Priority filter
	O	Generates requests with a Notification Class filter
	O	Generates requests with multiple filters
	¹ This BIBB can be claimed if AE-AS-A is also claimed.	
Alarm and Event Management - Enrollment Summary - B		
	R	Base Requirements
Alarm and Event Management - Information - A		
	R¹	Base Requirements
	¹ This BIBB can be claimed if AE-AS-A is also claimed.	
...		

Test Plan Changes

[Remove Clause 5.6 Alarm and Event Management - Alarm Summary – A]
[Remove Clause 5.8 Alarm and Event Management - Enrollment Summary – A]
[Remove Clause 5.10 Alarm and Event Management - Information - A]

5.20 Alarm and Event Management - Alarm Summary View - A

5.20.3 Supports Initiation of GetAlarmSummary Service

The IUT shall support *GetAlarmSummary service* AE-ASUM-A in order to update alarm parameters modified by the user.

135.1-2023 - 8.6.1 - Basic GetAlarmSummary Service Initiation		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	

Specified Test Changes

[Remove 135.1-2023, Clause 8.7 including all tests]

BTL-26.0 imp1-7: Add NM-FDR-A to IPv4 and IPv6 Datalinks [BTLWG-1586]

Overview:

In Test Plan Clauses 9.3.3 and 9.8.3 Add Verify Checklist NM-FDR-A.

Changes:

Checklist Changes

[Remove Supports NM-BBMDC-B from Data Link Layer - IPv4 and renumber footnotes.]

9   Data Link Layer

Data Link Layer - IPv4		
	R	Base Requirements
	C[1]	Is able to operate in Normal mode
	C[1]	Is able to operate in Foreign mode
	C[1]	Is able to operate in BBMD mode
	C[2]	Supports the Network Port object
	O	Is able to initiate broadcast messages
	O	Supports Network Port objects and DHCP
	O	Supports Network Address Translation in BBMD mode
	~~BTL-C[3]~~	~~Supports NM-BBMDC-B~~
	O[3]	Supports configurable Out_Of_Service property
	C[4]	Supports hierarchical Network Port objects
	C[4]	Supports Non-hierarchical Network Port objects
	C[3,5]	Supports writable Network_Number property
	O[3]	Supports the Routing_Table property
	O[3]	Supports the Network Port Object Command property
	O[3,6]	Supports the DISCARD_CHANGES command
	O[3,6]	Supports the RENEW_FD_REGISTRATION command
	O[3,6]	Supports the RENEW_DHCP command
	O[3,6]	Supports the RESTART_AUTONEGOTIATION command
	O[3,6]	Supports the RESTART_PORT command
	O[6,7]	Supports the VALIDATE_CHANGES command
[1] Either BBMD or both Normal and Foreign modes are required.		
[2] This option is required if the IUT claims Protocol_Revision 17 or higher.		
[3] ~~Required if the device is able to operate in BBMD mode.~~		
[3] Protocol_Revision 17 or higher must be claimed.		
[4] At least one of these options is required if the IUT claims Protocol_Revision 17 or higher.		
[5] Support for writable Network_Number properties is required in routers and other IUTs that need to know the network number in order to operate.		
[6] At least one of these options is required if the Network Port object Command property is supported.		
[7] Protocol_Revision 24 or higher must be claimed.		

Test Plan Changes

[In Data Link Layer - IPv4, remove section 9.3.9 Supports NM-BBMDC-B and renumber all sections below.]

[In section 9.3.3 - Is Able to Operate in Foreign Mode, add this Verify Checklist.]

Verify Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify the IUT claims support for NM-FDR-A
	Testing Hints	

[In section 9.3.4 - Is Able to Operate in BBMD Mode, add this Verify Checklist.]

Verify Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify the IUT claims support for NM-BBMDC-A
	Testing Hints	

[In section 9.8.3 - Is Able to Operate in Foreign Mode, add this Verify Checklist.]

Verify Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify the IUT claims support for NM-FDR-A
	Testing Hints	

[In section 9.8.4 - Is Able to Operate in BBMD Mode, remove this Verify Checklist.]

~~Verify Checklist~~		
	~~Test Conditionality~~	~~Must be executed.~~
	~~Test Directives~~	~~Verify the IUT claims support for NM-BBMDC-A~~
	~~Testing Hints~~	

[In section 9.9.3 - Is Able to Operate as a Hub, add this Verify Checklist.]

Verify Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify the IUT claims support for NM-SCHub-B
	Testing Hints	

[In section 9.9.5 - Is Able to Accept Direct Connections, add this Verify Checklist.]

Verify Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify the IUT claims support for NM-SCDC-B
	Testing Hints	

[In section 9.9.6 - Is Able to Initiate Direct Connections, add this Verify Checklist.]

Verify Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify the IUT claims support for NM-SCDC-A
	Testing Hints	

Specified Test Changes

None

BTL-26.0 imp1-8: EventLog for Non-transmitted Events [BTLWG-1595]

Overview:

The standard allows for event logs to contain events that were never transmitted by APDU. The requirements for such events are different from those transmitted by APDU (which always have an associated notification class object) so a different test is needed. (See IC, Oct 2024)

Changes:

Checklist Changes

[In BTL Checklist, add entries to AE-EL-I-B (section 5.14)]

Alarm and Event Management - Event Log - Internal - B		
	R	Base Requirements
	R	Supports all forms of ReadRange
	R	Executes ReadRange
	O	Supports logging of ACK_NOTIFICATION
	O	Supports Start_Time and Stop_Time properties
	C[1]	*Supports logging of events transmitted as an APDU*
	C[1]	*Supports logging of events never transmitted as an APDU*
[1]At least one of these options must be selected in order to claim this BIBB		

Test Plan Changes

[In BTL Test Plan, makes changes to section 5.14 as shown below]

5.14.1 Base Requirements

Base requirements must be met by any IUT claiming conformance to this BIBB.

...		
~~135.1-2023 - 7.3.2.25.1 - Internal Logging of Notifications~~		
	~~Test Conditionality~~	~~Must be executed~~
	~~Test Directives~~	~~REPEAT for both Confirmed and UnconfirmedEventNotifications.~~
	~~Testing Hints~~	~~REPEAT for events with optional message text present, and with message text not present if the device supports both.~~
...		

5.14.6 Supports Logging of Events Transmitted as an APDU

The IUT is able to generate an event notification sent to another device and record the event in an Event-Log's log buffer property.

~~135.1-2023~~*BTL* - 7.3.2.25.1 - Internal Logging of Notifications	
Test Conditionality	Must be executed.
Test Directives	REPEAT for both Confirmed and UnconfirmedEventNotifications.
Testing Hints	REPEAT for events with optional message text present, and with message text not present if the device supports both.

5.14.7 Supports Logging of Events Never Transmitted as an APDU

The IUT generates event notifications that are never transmitted as an APDU but are stored in an Event-Log's log buffer property.

BTL - 7.3.2.25.X1 - Logging Events Never Transmitted as an APDU

Test Conditionality	Must be executed.
Test Directives	
Testing Hints	

Specified Test Changes

[In BTL Specified Tests, add test 7.3.2.25.X1 and move 7.3.2.25.1 into BTL Specified Tests and change as shown]

7.3.2.25.X1 Logging Events Never Transmitted as an APDU
Reason for change: No tests exist for this functionality.

Purpose: To verify the IUT correctly stores event log records for event notifications that are never transmitted as an APDU.

Test Concept: The IUT is made to generate an event that is logged in the event log but is not intended for transmission as an APDU, such that the associated notification-class object is uninitialized (does not exist in the IUT). The event log record is read to verify it is stored as a proper BACnetEventLogRecord.

Configuration Requirements: Event Log, LO1 is configured to log the event notification(s) generated in the test.

Test Steps:

1. READ RC = LO1, Record_Count
2. MAKE (the device generate an event that gets entered into the log buffer of LO1)
3. TRANSMIT ReadRange-Request,
    'Object Identifier'=   LO1,
    'Property Identifier'= Log_Buffer,
    'Reference Index'=   RC+1,
    'Count'=   -1
4. RECEIVE ReadRange-ACK,
    'Object Identifier'=   LO1,
    'Property Identifier'= Log_Buffer,
    'Result Flags'=   {?,?,FALSE},
    'Item Count'=   1
    'Item Data'=
        'Timestamp'=   (any valid timestamp),
        'Log-datum'=   notification,
            'Process Identifier'=   (any valid value)
            'Initiating Device Identifier' =   IUT,
            'Event Object Identifier' =   (any valid object),
            'Time Stamp' =   (any valid timestamp),
            'Notification Class' =   (a value that does not correspond to a notification-class instance in the
IUT),
            'Priority' =   (any valid priority),
            'Event Type' =   (any standard event type),
            'Message Text' =   (optional, any valid message text),
            'Notify Type' =   ALARM | EVENT,
            'AckRequired' =   TRUE | FALSE,
            'From State' =   (any valid state for this event type),
            'To State' =   (any valid state for this event type),
            'Event Values' =   (any values appropriate to the event type)

7.3.2.25.1 Internal Logging of Notifications

Reason for change: Fixed minor errors

Purpose: To verify the IUT correctly collects and represents the Notifications which it initiates.

Test Concept: Make the IUT generate two event notification messages which the IUT logs. Use ReadRange to retrieve them from an Event Log and compare the two representations.

Configuration Requirements: The tester shall choose two events which are configured to be sent to the TD and to be placed into one of the IUT's Event Logs, LO1.

Notes to Tester: When the UnconfirmedEventNotification service is used instead of the ConfirmedEventNotification service, the TD shall skip the steps in which a BACnet-SimpleACK-PDU is sent.

Test Steps:
1. WRITE Enable = TRUE
2. MAKE (a condition exist that will cause the device to generate an event transition)
3. WAIT ~~D1~~ *Notification Fail Time*
4. *EV1 =* RECEIVE ConfirmedEventNotification-Request,
    'Process Identifier' =   (any valid process identifier),
    'Initiating Device Identifier' =   IUT,
    'Event Object Identifier' = (any valid object),
    'Time Stamp' =   (T1, any valid timestamp),
    'Notification Class' =   (any valid notification class),
    'Priority' =   (any valid priority),
    'Event Type' =   (any standard event type),
    'Message Text' =   (optional, any valid message text),
    'Notify Type' =   ALARM | EVENT,
    'AckRequired' =   TRUE | FALSE,
    'From State' =   (state S1, any valid state for this event type),
    'To State' =   (state S2, any valid state for this event type that can follow S1),
    'Event Values' =   (any values appropriate to the event type)
5. TRANSMIT BACnet-SimpleACK-PDU
6. MAKE (a condition exist that will cause the device to generate an event transition)
7. WAIT ~~D2~~ *Notification Fail Time*
8. *EV2 =* RECEIVE ConfirmedEventNotification-Request,
    'Process Identifier' =   (any valid process identifier),
    'Initiating Device Identifier' =   IUT,
    'Event Object Identifier' =   (any valid object),
    'Time Stamp' =   (T2, any valid timestamp),
    'Notification Class' =   (any valid notification class),
    'Priority' =   (any valid priority),
    'Event Type' =   (any standard event type),
    'Message Text' =   (optional, any valid message text),
    'Notify Type' =   ALARM | EVENT,
    'AckRequired' =   TRUE | FALSE,
    'From State' =   (state S3, any valid state for this event type),
    'To State' =   (state S4, any valid state for this event type that can follow S3),
    'Event Values' =   (any values appropriate to the event type)
9. TRANSMIT BACnet-SimpleACK-PDU
10. READ RC = LO1, Record_Count
11. TRANSMIT ReadRange-Request,
    'Object Identifier' =   LO1,
    'Property Identifier' = Log_Buffer,
    'Reference Index'=   RC,
    'Count'=   -2
12. RECEIVE ReadRange-ACK,
    'Object Identifier' =   LO1,
    'Property Identifier' = Log_Buffer,

        'Result Flags' =   {?, ?, FALSE},
        'Item Count' =   2,
        'Item Data' =   (logged data that matches *EV1 and EV2*, ~~the information received in steps 3 and 6,~~
                        except that Process_Identifier may be any value and is not required to match)

13. CHECK (T2 > T1, and that the notifications were logged in order)

BTL-26.0 imp1-9: 7.3.1.1.1 - Reliability as of PR20 [BTLWG-1616]

Overview:

As of PR20, if Reliability can be any value other than NO_FAULT_DETECTED, it must be writable.
At any PR, if Reliability can be any value other than NO_FAULT_DETECTED, add steps to test it.

Changes:

Checklist Changes

None

Test Plan Changes

None

Specified Test Changes

7.3.1.1.1 Out_Of_Service, Status_Flags, and Reliability Test

Reason for Change: Modified test to be more automation friendly. *Add PR 20 requirements and test that a change to the Present_Value that results in a change to Reliability is also tested.*

Purpose: To verify that Present_Value is writable when Out_Of_Service is TRUE and that the interrelationship between the Out_Of_Service, Status_Flags, and Reliability properties.

Test Concept: The value of the Out_Of_Service property is set to TRUE and the Present_Value property is tested to be writable. *If a value of the Present_Value results in a change to the Reliability property, verify the Status_Flags change. If the Reliability property is present and writable, verify a write to the Reliability property changes the Status Flags.* ~~The value of the Status_Flags property is validated and, if present, the value of the Reliability property is also validated.~~ The value of the Status_Flags property, ~~SF1,~~ and, if present, the Reliability property, ~~R1,~~ are checked to ensure they return to their initial values when the value of the Out_Of_Service property is set to FALSE.

Configuration Requirements: If the selected object is commandable, the values of the entries in the Priority_Array above the selected priority, PTY1, shall be NULL. *The Reliability property shall be NO_FAULT_DETECTED.*

Test Steps:

1. ~~READ SF1 = Status_Flags~~
1. *READ PV1 = Present_Value*
2. *VERIFY Status_Flags = (?, FALSE, ?, FALSE)*
2. ~~IF Reliability is present THEN~~
4. ~~READ R1 = Reliability~~
3. IF (Out_Of_Service is writable) THEN
4.   WRITE Out_Of_Service = TRUE
    ELSE
5.   MAKE (Out_Of_Service TRUE)
6. VERIFY Out_Of_Service = TRUE
7. VERIFY Status_Flags = (?, *FALSE*, ?, TRUE)
8. REPEAT X = (all values meeting the functional range requirements of 7.2.1) DO {
9.   WRITE Present_Value, PTY1 = X
10.   VERIFY Present_Value = X
11.   *IF Reliability is present THEN*

-- Check if Reliability changed due to a change to the Present Value
12.   IF (Reliability <> NO_FAULT_DETECTED) THEN
13.   VERIFY Status Flags = (?, TRUE, ?, TRUE)
14.   WRITE Present_Value, PTY1 = PV1
15.   VERIFY Reliability = NO_FAULT_DETECTED
16.   VERIFY Status_Flags = (?, FALSE, ?, TRUE)
    }
17. WRITE Present_Value, PTY1 = PV1
18. *IF (Reliability is present) THEN*
19.   *IF (((Protocol_Revision >= 20) AND (Reliability can take on a value other than*
         NO_FAULT_DETECTED)) OR (Reliability is writable)) THEN
~~9.   IF (Reliability is present and writable) THEN~~
20.   REPEAT X = (all values of the Reliability enumeration *supported by* ~~appropriate to~~ the object type
          except NO_FAULT_DETECTED) DO {
21.   WRITE Reliability = X
22.   VERIFY Reliability = X
23.   VERIFY Status_Flags = (?, TRUE, ?, TRUE)
24.   WRITE Reliability = NO_FAULT_DETECTED
25.   VERIFY Reliability = NO_FAULT_DETECTED
26.   VERIFY Status_Flags = (?, FALSE, ?, TRUE)
      }
27. IF (Out_Of_Service is writable) THEN
28.   WRITE Out_Of_Service = FALSE
   ELSE
29.   MAKE (Out_Of_Service FALSE)
30. VERIFY Out_Of_Service = FALSE
31. VERIFY Status_Flags = ~~SF1~~*(?, FALSE, ?, FALSE)*
32. IF Reliability is present THEN
33.   VERIFY Reliability = ~~R1~~*NO_FAULT_DETECTED*

BTL-26.0 imp1-10: Change of Value Notification Test Name Changes [BTLWG-1622]

Overview:

A simpler way to distinguish between the tests. We are already doing it that way in chapter 4.10.29.

Changes:

Checklist Changes

None

Test Plan Changes

[Reason for change: A simpler way to distinguish the tests]

4.10.35 Supports COV for Other Standard Object Types

The IUT accepts COV subscriptions and initiate COV notifications for other standard objects.

BTL - 8.2.X2 - Change of Value Notification from Other Standard Object Types *(ConfirmedCOVNotification)*	
Test Conditionality	This test may be skipped if 8.3.X2 is executed.
Test Directives	Execute this test with each standard object type not specified in 135-2020 Table 13-1 that supports COV.
Testing Hints	
BTL - 8.3.X2 - Change of Value Notification from Other Standard Object Types *(UnconfirmedCOVNotification)*	
Test Conditionality	This test may be skipped if 8.2.X2 is executed.
Test Directives	Execute this test with each standard object type not specified in 135-2020 Table 13-1 that supports COV.
Testing Hints	

Specified Test Changes

[Reason for change: A simpler way to distinguish the tests. No change to the actual tests.]

8.2.X2 Change of Value Notification from Other Standard Object Types*(ConfirmedCOVNotification)*

8.3.X2 - Change of Value Notification from Other Standard Object Types (UnconfirmedCOVNotification)

BTL-26.0 imp1-11: Timer Accepts all the Required Datatypes in an Internal Reference [BTLWG-1655]

Overview:

ANSI/ASHRAE Standard 135-2020 Clause 12.57 states: "If the properties State_Change_Values, List_Of_Object_Property_References and Priority_For_Writing are present and writable, then the Timer object shall be capable of writing values of type NULL, BOOLEAN, Unsigned, INTEGER, REAL, and ENUMERATED to properties in the local device referenced by List_Of_Object_Property_References. Support for writing to properties in other devices is optional."

Therefore State_Change_Values is added to the description of 3.57.8

ANSI/ASHRAE Standard 135.1-2023 - 7.3.2.47.1.8 Timer Accepts all the Required Datatypes in an Internal Reference

Changes:

Checklist Changes

None

Test Plan Changes

Reason for change: State_Change_Values was missing for the required properties. The Test itself claims to repeat the test for the different datatypes therefore it is removed from "Test Directives".

3.57.8 Supports Writable Priority_For_Writing and List_Of_Object_Property_References

The IUT supports writable *State_Change_Values,* Priority_For_Writing and List_Of_Object_Property_References properties in a Timer object.

135.1-2023 - 7.3.2.47.1.8 - Timer Accepts all the Required Datatypes in an Internal Reference		
	Test Conditionality	Must be executed.
	Test Directives	This test shall be executed with a Timer object that can be configured to monitor a property within the IUT. ~~Repeat the test with the List_Of_Object_Property_References making its references to, and the State_Change_Values property containing non-NULL values of each of these datatypes: NULL, BOOLEAN, Unsigned, INTEGER, REAL, and ENUMERATED. Support for writing to properties with other datatypes is optional.~~
	Testing Hints	

Specified Test Changes

None

BTL-26.0 imp1-12: Add Timer Testing to Scheduling - View Modify - A [BTLWG-1704]

Overview:

SCHED-VM-A and SCHED-AVM-A do not allow the vendor to specify the data types supported and no Timer object specific tests are executed. Null value and no-value Null needs to be tested in the State_Change_Values property.

Changes:

Checklist Changes

[Modify the Scheduling – View Modify - A section]

Scheduling - Advanced View Modify - A		
	R	Base Requirements
	R	Supports SCHED-VM-A
	R	Supports DM-OCD-A
Scheduling - View Modify - A		
	R	Base Requirements
	R	Supports DS-RP-A
	R	Supports DS-WP-A
	R	Is able to schedule any B-side device regardless of the claimed Protocol_Revision in the B-side device.
	R	Is able to present and modify schedules that do not include the Exception_Schedule property
	R	Is able to present and modify a Calendar Object
	R	Is able to present and modify schedules of Enumerated type
	R	Is able to present and modify schedules of REAL type
	R	Is able to present and modify schedules of Unsigned32 type
	R^1	Is able to present and modify schedules of BOOLEAN type
	O	Is able to present and modify schedules of INTEGER (signed) type
	O	Is able to present and modify schedules of Double type
	O	Is able to present and modify schedules of Octet String type
	O	Is able to present and modify schedules of Character String type
	O	Is able to present and modify schedules of Bit String type
	O	Is able to present and modify schedules of Date type
	O	Is able to present and modify schedules of Time type
	O	Is able to present and modify schedules of BACnetObjectIdentifier type
	C^2	*Is able to present and modify timers of Enumerated type*
	C^2	*Is able to present and modify timers of Real type*
	C^2	*Is able to present and modify timers of Unsigned type*
	C^2	*Is able to present and modify timers of Boolean type*
	O^3	*Is able to present and modify timers of Integer (signed) type*
	O^3	*Is able to present and modify timers of Double type*
	O^3	*Is able to present and modify timers of OctetString type*
	O^3	*Is able to present and modify timers of CharacterString type*
	O^3	*Is able to present and modify timers of BitString type*
	O^3	*Is able to present and modify timers of DatePattern type*
	O^3	*Is able to present and modify timers of TimePattern type*
	O^3	*Is able to present and modify timers of BACnetObjectIdentifier type*
	O^3	*Is able to present and modify timers of Constructed values.*
	O^3	*Is able to present and modify timers of BACnetDateTime type*
	O^3	*Is able to present and modify timers of BACnetLightingCommand type*

> [1] BTL-R if the IUT claims a revision before Protocol_Revision 20.
> [2] *Required if the IUT claims Protocol_Revision 17 or higher.*
> [3] *Protocol_Revision 17 or higher must be claimed.*

Test Plan Changes

6.1 Scheduling - Advanced View Modify - A

6.1.1 Base Requirements

Base requirements must be met by any IUT claiming conformance to this BIBB.

135.1-2023 - 8.18.3 - Reading and Presenting Properties		
	Test Conditionality	Must be executed.
	Test Directives	*For a Schedule and Calendar object, repeat the test for each property specified in Table K-25 of the 135 standard.* ~~Repeat the test for each of the standard object types, except Timer object for devices claiming Protocol_Revision 16 and below, and associated properties specified in Table K-20 in the 135 standard.~~ The reference schedule used during this test should include an Exception_Schedule that contains 255 entries and contain 12 BACnetTimeValue tuples per entry. The reference schedule should also contain a Weekly_Schedule which contains 6 BACnetTimeValue tuples per day. The Calendar Date_List used in this test should contain 32 calendar entries. *If Protocol_Revision 17 or greater is claimed, repeat the test for each property specified in Table K-25 of the 135 standard for the Timer object.*
	Testing Hints	
135.1-2023 - 8.22.4 - Accepting Input and Modifying Properties		
	Test Conditionality	Must be executed.
	Test Directives	*For a Schedule and Calendar object, repeat the test for each property specified in Table K-25 of the 135 standard.* ~~Repeat the test for each of the standard object types, except Timer object for devices claiming Protocol_Revision 16 and below, and associated properties specified in Table K-20 in the 135 standard.~~ The reference schedule used during this test should include an Exception_Schedule that contains 255 entries and contain 12 BACnetTimeValue tuples per entry. The reference schedule should also contain a Weekly_Schedule which contains 6 BACnetTimeValue tuples per day. The Calendar Date_List used in this test should contain 32 calendar entries. *If Protocol_Revision 17 or greater is claimed, repeat the test for each property specified in Table K-25 of the 135 standard for the Timer object.*
	Testing Hints	
…		

6.2 Scheduling - View Modify - A

6.2.1 Base Requirements

Base requirements must be met by any IUT claiming conformance to this BIBB.

135.1-2023 - 8.18.3 - Reading and Presenting Properties		
	Test Conditionality	If the IUT claims SCHED-AVM-A, this test may be omitted otherwise, this test must be executed.
	Test Directives	*For a Schedule and Calendar object, repeat the test for each property specified in Table K-26 of the 135 standard.* ~~Repeat the test for each of the standard object types, defined in the claimed Protocol_Revision, and associated properties specified by SCHED-VM-A.~~ The reference schedule used during this test should include an Exception_Schedule that contains 255 entries and contain 12 BACnetTimeValue tuples per entry. The reference schedule should also contain a Weekly_Schedule which contains 6 BACnetTimeValue tuples per day. The Calendar Date_List used in this test should contain 32 calendar entries. *If Protocol_Revision 17 or greater is claimed, repeat the test for each property specified in Table K-26 of the 135 standard for the Timer object.*
	Testing Hints	
135.1-2023 - 8.22.4 - Accepting Input and Modifying Properties		
	Test Conditionality	If the IUT claims SCHED-AVM-A, this test may be omitted otherwise, this test must be executed.
	Test Directives	*For a Schedule and Calendar object, repeat the test for each property specified in Table K-26 of the 135 standard.* ~~Repeat the test for each of the standard object types, defined in the claimed Protocol_Revision, and associated properties specified by SCHED-VM-A.~~ The reference schedule used during this test should include an Exception_Schedule that contains 255 entries and contain 12 BACnetTimeValue tuples per entry. The reference schedule should also contain a Weekly_Schedule which contains 6 BACnetTimeValue tuples per day. The Calendar Date_List used in this test should contain 32 calendar entries. *If Protocol_Revision 17 or greater is claimed, repeat the test for each property specified in Table K-26 of the 135 standard for the Timer object.*
	Testing Hints	

<mark>[Add new Clauses to 6.2]</mark>

6.2.X1 Is able to Present and Modify timers of Enumerated Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	

6.2.X2 Is able to Present and Modify timers of Real Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
Test Directives	
Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
Test Directives	
Testing Hints	

6.2.X3 Is able to Present and Modify timers of Unsigned Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
Test Directives	
Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
Test Directives	
Testing Hints	

6.2.X4 Is able to Present and Modify timers of BooleanType

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
Test Directives	
Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
Test Directives	
Testing Hints	

6.2.X5 Is able to Present and Modify timers of Integer (Signed) Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
Test Directives	
Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.

Test Directives	
Testing Hints	

6.2.X6 Is able to Present and Modify timers of Double Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
Test Directives	
Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
Test Directives	
Testing Hints	

6.2.X7 Is able to Present and Modify timers of OctetString Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
Test Directives	
Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
Test Directives	
Testing Hints	

6.2.X8 Is able to Present and Modify timers of CharacterString Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
Test Directives	
Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property	
Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
Test Directives	
Testing Hints	

6.2.X9 Is able to Present and Modify timers of BitString Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	

6.2.X10 Is able to Present and Modify timers of DatePattern Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	

6.2.X11 Is able to Present and Modify timers of TimePattern Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	

6.2.X12 Is able to Present and Modify timers of BACnetObjectIdentifier Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.

	Test Directives	
	Testing Hints	

6.2.X13 Is able to Present and Modify timers of Constructed Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	

6.2.X14 Is able to Present and Modify timers of BACnetDateTime Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	

6.2.X15 Is able to Present and Modify timers of BACnetLightingCommand Type

The IUT can read, present and modify timers that contain State_Change_Values and List_Of_Object_Property_References properties with values of the specified data type.

BTL - 13.10.X.1 - Read and Present a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	
BTL - 13.10.X.2 - Modify a State_Change_Values Property		
	Test Conditionality	The reference server timer (T1) shall contain values of the specified data type.
	Test Directives	
	Testing Hints	

Specified Test Changes

[Modify 135.1-2023 Clause 13.10]
13.10 Workstation Scheduling Tests

Purpose: This group of tests verifies that the IUT is capable of viewing and modifying existing ~~schedules,~~ *Schedule, Calendar and Timer objects*.

Test Concept: This test consists of high-level MAKE and CHECK steps that are expected to be manually executed while monitoring BACnet communications using a BACnet network analyzer.

Configuration Requirements: The reference device shall be configured to indicate that it supports only the ReadProperty-Request and WriteProperty-Request services in the Protocol_Services_Supported property of its Device object. (Service clients that can use services more complex than ReadProperty-Request, such as ReadPropertyMultiple-Request, shall be able to adapt to a server device that does not support these more complex services.) The reference device is configured to contain Schedules, *Calendars, and Timers* ~~and Calendars~~ as specified by S1, S2, C1, *C2, and T1* ~~and C2~~. The datatype of the 'value' portion of BACnetTimeValue can be varied to test scheduling of different datatypes, but all of the BACnetTimeValue elements shall be consistent within the same Schedule object, and the Present_Value and properties referenced by List_Of_Object_Property_References shall also be of the same datatype. The reference objects S1, S2, C1, *C2, and T1* ~~and C2~~ represent the standard test data, but the tester is free to use additional test data for this test.

~~Note: The reference Schedule and Calendars contain some data that requires support for Protocol_Revision 4 or later. To convert the Schedule to conform to Protocol_Revision 3 or older, make the following changes:~~

~~1. Change month 13 to month 3 in BACnetSpecialEvent[5].~~
~~2. Remove the Schedule_Default property.~~
~~3. Change month 14 to month 2 in BACnetSpecialEvent[6].~~
~~4. Change dayOfWeek from 32 to 28 in BACnetSpecialEvent[6].~~
~~5. Change the last two CalendarEntries in the Date_List of Calendar C1 to remove the use of special values indicating all even months, all odd months, and the Last Day of the month.~~

…
[Add to the end of Clause 13.10]
Reference Timer T1:

> *State_Change_Values = {*
> > *<value1>,*
> > *Null, -- indicates Relinquish*
> > *<value2>,*
> > *<value3>,*
> > *[1] Null, -- indicates no-value*
> > *<value4>,*
> > *<value5>}*

[Add a new tests into BTL Specified Tests]

13.10.X Modify a Timer Object

This clause is used to verify that the IUT can present and modify a Timer object in a server device.

13.10.X.1 Read and Present a State_Change_Values Property

Purpose: Demonstrate that the IUT reads and presents the State_Change_Values property of a Timer object.

Configuration Requirements: A reference device contains Timer object T1.

Test Steps:

1. MAKE (the IUT read and present the data represented by the State_Change_Values of T1)
2. CHECK (Did the IUT properly present the data represented by the State_Change_Values?)

13.10.X.2 Modify a State_Change_Values Property

Purpose: Demonstrate that the IUT can accept user input and use it to modify the State_Change_Values and List_Of_Object_Property_References properties.

Configuration Requirements: A reference device contains a Timer object (T1) with writable State_Change_Values and List_Of_Object_Property_References properties.

Test Steps:

1. MAKE (the IUT write the State_Change_Values property with a set of values of data type DT1.
    Ensure at least one value is Null and one is no-value)
2. MAKE (the IUT write List_Of_Object_Property_References with a reference that accepts data type DT1)
3. CHECK (Did the IUT write the change to the State_Change_Values and List_Of_Object_Property_References
    properties correctly?)