



BACnet® TESTING LABORATORIES

Revision 26.1 Final
SPECIFIED TESTS

Revised November 8, 2025

Table of Contents

1. PURPOSE	4
7. OBJECT SUPPORT TESTS	5
7.2 Write Support for Properties in the Test Database.....	5
7.2.9 DateTime Non-Pattern Properties Test.....	5
7.3 Object Functionality Tests	5
7.3.1 Property Tests.....	5
7.3.1.1 Out_Of_Service, Status_Flags, and Reliability Tests.....	5
7.3.1.1.1 Out_Of_Service, Status_Flags, and Reliability Test	5
7.3.1.11.4 Acked_Transitions Test for Normal to Fault Transitions	6
7.3.1.21 Reliability_Evaluation_Inhibit Tests	8
7.3.1.21.1 Reliability_Evaluation_Inhibit Test.....	8
7.3.1.22 Event_Detection_Enable Tests.....	9
7.3.1.22.1 Event_Detection_Enable Inhibits Event Generation	9
7.3.1.28 Value Source Mechanism Tests.....	10
7.3.1.28.1 Writing to the Value_Source Property by a Device Other than the Device that Commanded the Object	10
7.3.1.28.2 Non-commandable Value_Source Property Test.....	11
7.3.1.28.3 Value_Source Property None Test.....	12
7.3.1.28.4 Commandable Value Source Test.....	12
7.3.1.28.5 Life Safety Value_Source Property Test.....	13
7.3.1.28.X1 Value Source Initiated Locally Test.....	13
7.3.1.X73 Number_Of_States Property Tests	14
7.3.1.X73.1 Writable Number_Of_States Test.....	14
7.3.2 Object Specific Tests.....	15
7.3.2.21 Notification Class Object.....	15
7.3.2.21.3 Recipient_List Tests.....	15
7.3.2.21.3.1 ValidDays Test.....	15
7.3.2.23 Schedule Object Tests.....	16
7.3.2.23.8 List_Of_Object_Property_Reference External Test	16
7.3.2.25 Event Log Tests	17
7.3.2.25.1 Internal Logging of Notifications	17
7.3.2.25.X1 Logging Events Never Transmitted as an APDU	18
7.3.2.30 Notification Forwarder Object Tests	19
7.3.2.30.13 Persistence Tests	19
7.3.2.30.13.1 Recipient_List Persistence Test	19
7.3.2.30.13.2 Subscribed_Recipients Persistence Test	20
7.3.2.39 Lighting Output Object Tests	21
7.3.2.39.5 Lighting Command Operation RAMP_TO Test.....	21
7.3.2.41 Access Point Object Tests	22
7.3.2.41.4 Authorization Mode Test.....	22
7.3.2.46 Network Port Object Tests.....	24
7.3.2.46.3 Network Port Command Tests	24
7.3.2.46.3.2 DISCARD_CHANGES Command Tests	24
7.3.2.46.3.2.2 DISCARD_CHANGES Command Failure Test.....	24
7.3.2.46.3.6 RESTART_AUTONEGOTIATION Command Tests.....	25
7.3.2.46.3.6.2 RESTART_AUTONEGOTIATION Command Failure Test.....	25
7.3.2.46.4 Hierarchical Network Port Tests.....	25
7.3.2.46.X Valid Hierarchy Tests	26
7.3.2.46.X.1 Valid Hierarchy Chain - MSTP Test.....	26
7.3.2.46.X.2 Valid Hierarchy Chain - IPvX Test.....	26
7.3.2.46.X.3 Valid Hierarchy Chain - Application and Physical Test	27
7.3.2.46.X.4 Valid Hierarchy Chain - Secure Connect Test.....	28
7.3.2.46.X.5 Valid Hierarchy Chain - Proprietary Test.....	29
7.3.2.46.X.Y Valid Hierarchy Chain - Virtual Test.....	30

7.3.2.47	Timer Object.....	31
7.3.2.47.1	Positive Tests	31
7.3.2.47.1.12	Forcing Timer Expiration by Writing IDLE	31
7.3.2.50	Staging Object Tests	32
7.3.2.50.11	Out_Of_Service, Status_Flags, and Reliability for Staging Object	32
8.	APPLICATION SERVICE INITIATION TESTS	33
8.2	ConfirmedCOVNotification Service Initiation Tests.....	33
8.2.1	Change of Value Notification for Changes to Present_Value in Objects with a COV_Increment.....	33
8.2.19	Change of Value Notification from Other Standard Object Types (ConfirmedCOVNotification)	
	34	
8.3	UnconfirmedCOVNotification Service Initiation Tests	35
8.3.21	- Change of Value Notification from Other Standard Object Types (UnconfirmedCOVNotification)	35
8.4	ConfirmedEventNotification Service Initiation Tests	36
8.4.14	UNSIGNED_RANGE Test (ConfirmedEventNotification Test).....	36
8.7	GetEnrollmentSummary Service Initiation Tests	38
8.21	ReadRange Service Initiation Tests.....	38
8.21.9	Presents Log Records	38
8.45	You-Are Service Initiation Tests	38
8.45.1	Configures Other Device's MAC Address.....	38
9.	APPLICATION SERVICE EXECUTION TESTS	39
9.1	AcknowledgeAlarm Service Execution Tests.....	39
9.1.2	Negative AcknowledgeAlarm Service Execution Tests	39
9.1.2.1	Unsuccessful Alarm Acknowledgment of Confirmed Event Notifications Because the 'Time Stamp' is Too Old.....	39
9.2	ConfirmedCOVNotification Service Execution Tests.....	42
9.2.2	Negative ConfirmedCOVNotification Execution Tests	42
9.2.2.1	Change of Value Notification Arrives after Subscription has Expired	42
9.16	CreateObject Service Execution Tests	43
9.16.2	Negative CreateObject Service Execution Tests	43
9.16.2.3	Attempting to Create an Object with an Object Identifier That is Not Creatable by Specifying the Object Identifier.....	43
9.16.2.8	Attempting to Create a non-Supported Object Type (by Object Identifier).....	43
9.18	ReadProperty Service Execution Tests.....	44
9.18.1	Positive ReadProperty Service Execution Tests	44
9.18.1.6	Respects max-segments-accepted bit pattern.....	44
9.20	ReadPropertyMultiple Service Execution Tests	44
9.20.1	Positive ReadPropertyMultiple Service Execution Tests	44
9.20.1.16	ReadPropertyMultiple Array Properties	44
9.22	WriteProperty Service Execution Tests.....	45
9.22.1	Positive WriteProperty Service Execution Tests	45
9.22.1.6	Writing NULL to Non-commandable Properties.....	45
9.22.2	Negative WriteProperty Service Execution Tests.....	46
9.22.2.1	Writing Non-Array Properties with an Array Index	46
9.23	WritePropertyMultiple Service Execution Tests.....	47
9.23.2	Negative WritePropertyMultiple Service Execution Tests.....	47
9.23.2.14	Writing First Element of 'List of Write Access Specifications' with Object Access Error.....	47
9.23.2.15	Writing First Element of 'List of Write Access Specifications' with a Write Access Error	48
9.23.2.16	WritePropertyMultiple Reject Test for First Element of 'List of Write Access Specifications'.....	48
9.23.2.17	Writing First Element of 'List of Write Access Specifications' with a Property Access Error.....	49
9.23.2.21	DateTime Non-Pattern Properties Test using WritePropertyMultiple Service.....	49
9.33	Who-Is Service Execution Tests.....	50
9.33.2	Execution of Who-Is Service Requests Originating from a Remote Network.....	50

BACnet Testing Laboratories - Specified Tests

9.33.2.1	General Inquiry, Global Broadcast from a Remote Network	50
9.33.2.2	General Inquiry, Remote Broadcast.....	51
9.33.2.3	General Inquiry, Directed to a Remote Device.....	51
9.39	General Testing Service Execution	52
9.39.2	Unsupported Unconfirmed Services Test	52
10	NETWORK LAYER PROTOCOL TESTS	53
10.2	Router Functionality Tests	53
10.2.2	Processing Network Layer Messages	53
10.2.2.4.3	Receiving Messages for a Busy Router	53
10.7	Route Binding Tests.....	53
10.7.3	Router Binding via Who-Is-Router-To-Network	53
12	DATA LINK LAYER PROTOCOLS TESTS	56
12.3	BACnet/IP Functionality Tests	56
12.3.2	BBMD B/IP Device with a Server Application.....	56
12.3.2.2	Execute Original-Broadcast-NPDU.....	56
12.3.2.2.2	Execute Original-Broadcast-NPDU (Two-hop Distribution)	56
12.3.6	Foreign Device Management.....	57
12.3.6.3	Foreign Device Table Timer Operations	57
12.3.6.3.1	Non-Zero-Duration Foreign Device Table Timer Operations	57
12.4	BACnet/IPv6 Functionality Tests.....	58
12.4.4	BBMD Tests	58
12.4.4.1	Positive Tests	59
12.4.4.1.1	Original-Broadcast-NPDU.....	59
12.4.4.1.4	Forwarded-Address-Resolution	60
12.4.5	Foreign Device Management Tests	61
12.4.5.1	Execute Register-Foreign-Device.....	61
12.5	Secure Connect Functionality Tests.....	62
12.5.1	Basic Node Tests	62
12.5.1.1	Basic Node Positive Tests	62
12.5.1.1.16	Heartbeat-Request Initiation Test	62
12.5.2	Hub Tests.....	63
12.5.2.1	Hub Positive Tests	63
12.5.2.1.9	Duplicate Connection Test.....	63
12.5.2.1.11	SC_Hub_Function_Enable Property Test.....	65
12.5.3	Direct Connect Tests	67
12.5.3.3	Initiating Direct Connect Tests	67
12.5.3.3.2	Initiating Direct Connect Negative Tests.....	67
12.5.3.3.2.3	Rejection of Invalid Certificate Outgoing Connection Test.....	67
13	SPECIAL FUNCTIONALITY TESTS	67
13.10	Workstation Scheduling Tests	67
13.10.X	Modify a Timer Object.....	68
13.10.X.1	Read and Present a State_Change_Values Property	68
13.10.X.2	Modify a State_Change_Values Property.....	68

1. PURPOSE

This document contains tests defined by the BTL that are not included in ANSI/ASHRAE Standard 135.1-2025 or are modified versions of tests in 135.1. These tests are used by the BTL testing process and are referenced by the BTL Test Plan document.

Most of the tests defined in this document will be submitted to SSPC 135. Those that are submitted will be removed from future versions of this document as they are accepted/rejected by the SSPC 135 and 135.1 is updated.

Some of the tests are interim tests defined by the BTL because the test tools are not adequate for testing the particular functionality. These tests will be removed once the tests in SSPC 135.1 can be implemented by the BTL. Examples of such tests are the MS/TP tests.

7. OBJECT SUPPORT TESTS

7.2 Write Support for Properties in the Test Database

7.2.9 DateTime Non-Pattern Properties Test

Reason For Change: Remove day of week test based on IR.

Purpose: To verify that the property being tested does not accept special date field values.

Test Concept: The property being tested, P1, is written with each of the special datetime field values to ensure that the property does not accept them. A datetime DT1 is selected which is within the range that the IUT will accept for the property. The value, V1, written to the property, is the datetime DT1 with one of its fields replaced with one of the date or time special values. If the property is a complex datatype, the other fields in the value shall be set within the range accepted by the IUT. *It is a local matter whether the device accepts or rejects an invalid day of week field so it is not tested. This test shall only be applied to devices claiming Protocol_Revision 11 or higher.*

~~Notes to Tester: If P1 is an array, then an array index shall be provided in the TRANSMIT portion of step 1.~~

Test Steps:

1. REPEAT SV = (year unspecified, month unspecified, day of month unspecified, ~~day of week unspecified,~~ odd months, even months, last day of month, even days, odd days, hour unspecified, minute unspecified, second unspecified, hundredths unspecified) DO {
- +2. TRANSMIT WriteProperty-Request
 'Object Identifier' = O1,
 'Property Identifier' = P1,
 'Property Value' = (DT1 updated with the special value SV)
23. RECEIVE BACnet-Error-PDU
 'Error Class' = PROPERTY,
 'Error Code' = VALUE_OUT_OF_RANGE
 }

7.3 Object Functionality Tests

7.3.1 Property Tests

7.3.1.1 Out_Of_Service, Status_Flags, and Reliability Tests

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 = R1NO_FAULT_DETECTED

```

7.3.1.11.4 Acked_Transitions Test for Normal to Fault Transitions

Reason for Change: New test.

Purpose: To verify that the Acked_Transitions property tracks whether or not an acknowledgment has been received for a previously issued fault event notification. It also verifies the interrelationship between Status_Flags and Event_State.

Test Concept: The IUT is configured such that the Event_Enable property indicates that fault event transitions are to trigger an event notification. The Acked_Transitions property shall have the value (?, TRUE, ?). The fault event transition is triggered and the Acked_Transitions property is monitored to verify that the appropriate bit is cleared when a notification message is transmitted and reset when an acknowledgment is received.

Configuration Requirements: The Event_Enable and Acked_Transitions properties shall be configured with a value of (?, TRUE, ?). The referenced event-triggering property shall be set to a value that results in a NORMAL condition. The value of the Transitions parameter for all recipients shall be (?, TRUE, ?).

Notes to Tester: The UnconfirmedEventNotification service may be substituted for the ConfirmedEventNotification service, in which case the TD shall skip sending the BACnet-SimpleACK-PDU messages after receiving the notifications.

Test Steps:

1. VERIFY pCurrentState = NORMAL
2. VERIFY Acked_Transitions = (?, TRUE, ?)
3. IF (Protocol_Revision is present AND Protocol_Revision >= 13) THEN
4. VERIFY Status_Flags = (FALSE, FALSE, ?, ?)
5. MAKE (a condition exist that will cause the object to generate a fault condition)
6. **BEFORE Notification Fail Time**
7. RECEIVE ConfirmedEventNotification-Request,
 - 'Process Identifier' = (PI3: any valid process ID),
 - 'Initiating Device Identifier' = IUT,
 - 'Event Object Identifier' = (the event-generating object configured for this test),
 - 'Time Stamp' = (Tfault: any valid time stamp),
 - 'Notification Class' = (the class corresponding to the object being tested),
 - 'Priority' = (Pfault: the value configured to correspond to a TO-FAULT transition),
 - 'Event Type' = IF (Protocol_Revision < 13) THEN
 - (any valid event type),
 - ELSE
 - CHANGE_OF_RELIABILITY,
 - 'Message Text' = (optional, any valid message text),
 - 'Notify Type' = (the notify type configured for this event),
 - 'AckRequired' = TRUE,
 - 'From State' = NORMAL,
 - 'To State' = FAULT,
 - 'Event Values' = (values appropriate to the event type)
8. TRANSMIT BACnet-SimpleACK-PDU
9. VERIFY pCurrentState = FAULT
10. VERIFY Acked_Transitions = (?, FALSE, ?)
11. IF (Protocol_revision is present AND Protocol_Revision >= 13) THEN
12. VERIFY Status_Flags = (~~FALSE~~?, TRUE, ?, ?)
13. TRANSMIT AcknowledgeAlarm-Request,
 - 'Acknowledging Process Identifier' = (PI3),
 - 'Event Object Identifier' = (the event-generating object configured for this test),
 - 'Event State Acknowledged' = FAULT,
 - 'Acknowledgement Source' = (a character string),
 - 'Time Stamp' = (Tfault),
 - 'Time of Acknowledgment' = (the TD's current time)
14. RECEIVE BACnet-SimpleACK-PDU
15. IF (Protocol_Revision is present AND Protocol_Revision ≥ 1) THEN
16. **BEFORE Notification Fail Time**
17. RECEIVE ConfirmedEventNotification-Request,
 - 'Process Identifier' = (PI3),
 - 'Initiating Device Identifier' = IUT,


```

        'Event Object Identifier' = (the event-generating object configured for this test),
        'Time Stamp' = (Tfault the IUT's current time or sequence number),
        'Notification Class' = (the class corresponding to the object being tested),
        'Priority' = (Pfault),
        'Event Type' = IF (Protocol_Revision < 13)
                        (any valid event type),
                        ELSE
                        CHANGE_OF_RELIABILITY,
        'Message Text' = (optional, any valid message text),
        'Notify Type' = ACK_NOTIFICATION,
        'To State' = FAULT
    ELSE
18.    BEFORE Notification Fail Time
19.    RECEIVE ConfirmedEventNotification-Request,
        'Process Identifier' = (PI3),
        'Initiating Device Identifier' = IUT,
        'Event Object Identifier' = (the event-generating object configured for this test),
        'Time Stamp' = (Tfault the IUT's current time or sequence number),
        'Notification Class' = (the class corresponding to the object being tested),
        'Priority' = (Pfault),
        'Event Type' = (any valid event type),
        'Message Text' = (optional, any valid message text),
        'Notify Type' = ACK_NOTIFICATION
20.    TRANSMIT BACnet-SimpleACK-PDU
21.    VERIFY Acked_Transitions = (?, TRUE, ?)

```

7.3.1.21 Reliability_Evaluation_Inhibit Tests

7.3.1.21.1 Reliability_Evaluation_Inhibit Test

Reason for Change: Fixed step 20.

Purpose: To verify that Reliability_Evaluation_Inhibit controls whether or not fault conditions are detected and events are generated.

Test Concept: Select an event generating object, O1, which supports the Reliability_Evaluation_Inhibit property. With Reliability_Evaluation_Inhibit FALSE, make a fault condition exist. Verify that Reliability changes and that a notification is generated. Set Reliability_Evaluation_Inhibit to TRUE. Verify that the Reliability changes to NO_FAULT_DETECTED and that a TO_NORMAL notification is generated. Remove the fault condition and ensure that no notification is generated. Make a fault condition exist and verify that Reliability remains NO_FAULT_DETECTED, and that no notification is generated.

Configuration Requirements: O1 is configured to detect and report unconfirmed events, is in the NORMAL state, and Reliability_Evaluation_Inhibit equals FALSE, so that reliability evaluation for that object is configured to detect fault conditions.

Notes to Tester: This behavior can alternately be tested using the ConfirmedEventNotification service, but it is not necessary to test both.

Test Steps:

1. VERIFY pCurrentState = NORMAL
2. VERIFY Reliability = NO_FAULT_DETECTED
3. MAKE(a condition exist that would cause O1 to generate a TO_FAULT transition)
4. BEFORE **Notification Fail Time**

5. RECEIVE UnconfirmedEventNotification-Request
 - 'Process Identifier' = (the value configured for the transition),
 - 'Initiating Device Identifier' = IUT,
 - 'Event Object Identifier' = O1,
 - 'Time Stamp' = (any valid timestamp),
 - 'Priority' = (any valid priority),
 - 'Event Type' = CHANGE_OF_RELIABILITY,
 - 'Message Text' = (optional, any valid message text),
 - 'Notify Type' = ALARM | EVENT,
 - 'AckRequired' = TRUE | FALSE,
 - 'From State' = NORMAL,
 - 'To State' = FAULT,
 - 'Event Values' = (any values appropriate to CHANGE_OF_RELIABILITY)
6. VERIFY Reliability \Rightarrow NO_FAULT_DETECTED
7. VERIFY pCurrentState = FAULT
8. IF Reliability_Evaluation_Inhibit is writable THEN
9. WRITE Reliability_Evaluation_Inhibit = TRUE
- ELSE
10. MAKE(Reliability_Evaluation_Inhibit TRUE)
11. BEFORE **Internal Processing Fail Time + Notification Fail Time**
12. RECEIVE UnconfirmedEventNotification-Request
 - 'Process Identifier' = (the value configured for the transition),
 - 'Initiating Device Identifier' = IUT,
 - 'Event Object Identifier' = O1,
 - 'Time Stamp' = (any valid timestamp),
 - 'Priority' = (any valid priority),
 - 'Event Type' = CHANGE_OF_RELIABILITY,
 - 'Message Text' = (optional, any valid message text),
 - 'Notify Type' = ALARM | EVENT,
 - 'AckRequired' = TRUE | FALSE,
 - 'From State' = FAULT,
 - 'To State' = NORMAL,
 - 'Event Values' = (any values appropriate to CHANGE_OF_RELIABILITY)
13. VERIFY Reliability = NO_FAULT_DETECTED
14. VERIFY pCurrentState = NORMAL
15. MAKE(remove the fault condition)
16. WAIT(pTimeDelayNormal)
17. WAIT **Notification Fail Time**
18. CHECK (that the IUT did not send any event notifications for O1)
19. VERIFY Reliability = NO_FAULT_DETECTED
20. MAKE(Θ a condition exist that would cause O1 to generate a TO_NORMAL transition)
21. WAIT **Notification Fail Time**
22. VERIFY Reliability = NO_FAULT_DETECTED
23. VERIFY pCurrentState = NORMAL
24. CHECK (that the IUT did not send any event notifications for O1)

7.3.1.22 Event_Detection_Enable Tests

7.3.1.22.1 Event_Detection_Enable Inhibits Event Generation

Reason for Change: Updated purpose

Purpose: To verify that Event_Detection_Enable ~~enables and~~ disables event detection in objects which are configured for ~~event reporting~~

Test Concept: Select an event generating object, O1, that supports event reporting. Verify the Event_State is NORMAL and ~~the~~ Acked_Transitions, Event_Time_Stamps, and Event_Message_Texts are equal to their respective initial conditions, as mandated in the standard. Make a condition exist that would cause a transition if event reporting were enabled and observe that no notification messages are transmitted.

Configuration Requirements: O1 is configured with Event_Detection_Enable set to FALSE. If Event_Detection_Enable cannot be set to FALSE, this test shall be skipped. DELAY shall represent the time delay appropriate to the transition being tested (i.e. Time_Delay for TO_OFFNORMAL, 0 for TO_FAULT and FAULT to NORMAL transitions, and either Time_Delay or Time_Delay_Normal for TO_NORMAL). For this test, NO_TS equals a BACnetDateTime with all unspecified values, a BACnet Time with all unspecified values, or a sequence number of 0.

Notes to Tester: This behavior can alternately be tested using the ConfirmedEventNotification service, but it is not necessary to test both.

Test Steps:

1. VERIFY Event_Detection_Enable = FALSE
2. VERIFY pCurrentState = NORMAL
3. VERIFY Acked_Transitions = (T,T,T)
4. VERIFY Event_Time_Stamps = [NO_TS, NO_TS, NO_TS]
5. IF (Event_Message_Texts property exists) THEN
6. VERIFY Event_Message_Texts = ["", "", ""]
7. MAKE (a condition exist which would cause O1 to transition, if Event_Detection_Enable were TRUE)
8. WAIT (DELAY + **Notification Fail Time**)
9. CHECK (that the IUT did not send any event notifications for O1)
10. VERIFY pCurrentState = NORMAL
11. VERIFY Acked_Transitions = (T,T,T)
12. VERIFY Event_Time_Stamps = [NO_TS, NO_TS, NO_TS]
13. IF (Event_Message_Texts property exists) THEN
14. VERIFY Event_Message_Texts = ["", "", ""]

7.3.1.28 Value Source Mechanism Tests

7.3.1.28.1 Writing to the Value_Source Property by a Device Other than the Device that Commanded the Object

Reason for Change: Test does not check the value source properties for a commandable object. PRIO is not defined.

Purpose: To verify the IUT correctly refuses an attempt to write a Value_Source property by a device other than the device that most recently commanded the object.

Test Concept: Command an object, O1, that supports the value source mechanism, from device D1 (TD), and verify the updated Value_Source *and, if commandable, the other value source properties*. Attempt to write to the Value_Source property from device D2. Verify that an error is returned and Value_Source *and the other value source properties* ~~does~~ not change.

Configuration Requirements: *If commandable, the value selected for PRIO shall be numerically small enough to be the active priority.*

Notes to Tester: *The value of the network-number in the network address of the Value_Source property will be one of three values. It will be 0 if the TD and IUT are on the same network and the IUT does not know the network number. It will be the network number of the local network if the IUT knows the network number and the TD and IUT are on the same network. It will be the TD's network number if the TD is on a different network.*

Test Steps:

1. TRANSMIT WriteProperty-Request,
 SOURCE = D1,
 'Object Identifier' = O1,
 'Property Identifier' = (P1: the property monitored by the Value_Source mechanism for this object type),
 'Priority' = (~~absent or PRIO: absent or any value other than 6~~),
 'Property Value' = (X2: any valid value)
2. RECEIVE BACnet-SimpleACK-PDU
3. VERIFY Value_Source = (VS, D1's device identifier or network address)
- ~~43.~~ IF (O1 is commandable) THEN
 5. VERIFY Priority_Array = X2, ARRAY_INDEX = PRIO
 6. VERIFY Value_Source_Array = VS, ARRAY_INDEX = PRIO
 7. VERIFY Last_Command_Time ~= (T1, the current local time)
 8. IF (Command_Time_Array is present) THEN
 9. VERIFY Command_Time_Array = T1, ARRAY_INDEX = PRIO
- ELSE
 10. VERIFY (P1) = X2
- ~~4.~~ ~~VERIFY Value_Source = (D1's device identifier or network address)~~
 -- Attempt to write D1's device identifier from D2
- ~~115.~~ TRANSMIT WriteProperty-Request,
 SOURCE = D2,
 'Object Identifier' = O1,
 'Property Identifier' = Value_Source,
 'Priority' = PRIO,
 'Property Value' = (any valid value)
- ~~126.~~ RECEIVE BACnet-Error PDU,
 'Error Class' = PROPERTY,
 'Error Code' = WRITE_ACCESS_DENIED
 -- Verify value source properties are unchanged
- ~~137.~~ VERIFY Value_Source = (~~VSD1's device identifier or network address~~)
14. IF (O1 is commandable) THEN
 15. VERIFY Value_Source_Array = VS, ARRAY_INDEX = PRIO
 16. VERIFY Last_Command_Time = T1
 17. IF (Command_Time_Array is present) THEN
 18. VERIFY Command_Time_Array = T1, ARRAY_INDEX = PRIO

7.3.1.28.2 Non-commandable Value_Source Property Test

Reason for Change: Specify what is supposed to be written to the Value_Source property.

Purpose: To verify that the Value_Source property indicates the source of the current Present_Value in a non-commandable object.

Test Concept: Select a non-commandable object with a writable Present_Value which supports the Value Source mechanism. ~~The TD writes the Present_Value is written,~~ and it is verified that Value_Source is updated appropriately. Value_Source is then written to verify that the last writer, TD, can update it.

Notes to Tester: The value of the network-number in the network address of the Value_Source property will be one of three values. It will be 0 if the TD and IUT are on the same network and the IUT does not know the network number. It will be the network number of the local network if the IUT knows the network number and the TD and IUT are on the same network. It will be the TD's network number if the TD is on a different network.

Test Steps:

1. WRITE Present_Value = (V1, any valid value)
2. VERIFY Present_Value = V1

3. VERIFY Value_Source = (TD's device identifier or network address)
4. WRITE Value_Source = (*VS(TD's device identifier, O1, any valid object identifier, any valid value, V2)*)
5. VERIFY Value_Source = ~~VS~~~~V2~~

7.3.1.28.3 Value_Source Property None Test

Reason for Change: The other Value Source properties are not tested. The minimum on/off requirement from 19.5.1.3 paragraph 4 is not tested.

Purpose: To verify that the Value_Source property shall have the value 'None' when there is no active value source.

Test Concept: If there is no active value source, i.e., the Present_Value has taken on the value of Relinquish_Default, then the Value_Source property shall have the value 'None' *and the other value source properties are updated. If minimum on/off is supported verify the Value_Source_Array at priority 6 is commanded object, O1, and the Command_Time_Array has the present local time.*

Configuration Requirements: The object (O1) to be tested shall have 1 non-NULL entry in its Priority_Array and the Current_Command_Priority has a value other than NULL or 6.

Test Steps:

1. READ PRIO = Current_Command_Priority
2. ~~CHECK(PRIO < 6 and PRIO < NULL)~~
2. VERIFY Value_Source = (is not 'None')
3. WRITE Present_Value = NULL, PRIORITY = PRIO
4. VERIFY Last_Command_Time ~= (T1, the current local time)
5. *VERIFY Value_Source_Array = TD, ARRAY_INDEX = PRIO*
6. *IF (Command_Time_Array is present) THEN*
7. *VERIFY Command_Time_Array = T1, ARRAY_INDEX = PRIO*
8. *IF (O1 has Minimum_On_Time or Minimum_Off_Time properties) THEN*
9. *WAIT the larger of Minimum_Off_Time and Minimum_On_Time*
10. *VERIFY Value_Source_Array = O1, ARRAY_INDEX = 6*
11. *IF (Command_Time_Array is present) THEN*
12. *VERIFY Command_Time_Array ~= (the current local time), ARRAY_INDEX = 6*
13. VERIFY Current_Command_Priority = NULL
14. VERIFY Value_Source = 'None' -- the value is the choice 'none'

7.3.1.28.4 Commandable Value Source Test

Reason for Change: Command_Time_Array was not tested. Specify what is supposed to be written to the Value_Source property.

Purpose: To verify that the Value_Source, Value_Source_Array, and Last_Command_Time update correctly when Present_Value is written in a commandable object.

Test Concept: A commandable object which supports the value source mechanism is selected for the test. The Present_Value is written. Last_Command_Time, Value_Source, ~~and~~ Value_Source_Array, *and Command_Time_Array* properties are checked to verify that they have been updated appropriately. Value_Source is then written, and it is verified that Last_Command_Time property has not changed.

Configuration Requirements: The object being tested shall be commandable and support the Value Source mechanism. *The value selected for PRIO shall be numerically small enough to be the active priority. No other internal processes shall be controlling the object.*

Notes to Tester: The value of the network-number in the network address of the Value_Source property will be one of three values. It will be 0 if the TD and IUT are on the same network and the IUT does not know the network number. It will be the network number of the local network if the IUT knows the network number and the TD and IUT are on the same network. It will be the TD's network number if the TD is on a different network.

Test Steps:

- Verify that the value source properties are updated when Present_Value is commanded.
- 1. WRITE Present_Value = (V1: any valid value), PRIORITY = (PRIO:~~any value other than 6~~)
- 2. VERIFY Value_Source_Array = (VS~~SRC1~~: TD's device identifier or network address), ARRAY_INDEX = PRIO
- 3. VERIFY Value_Source = VS~~SRC1~~
- 4. VERIFY Last_Command_Time ~= (T1, the current local time)
- 5. IF (Command_Time_Array is present) THEN
- 6. VERIFY Command_Time_Array = T1, ARRAY_INDEX = PRIO
- Verify that Value_Source can be written and that Last_Command_Time does not update.
- ~~5. READ T1 = (O1), Last_Command_Time~~
- 7. WRITE Value_Source = (VS(TD's device identifier, O1, any valid object identifier-~~SRC2: any valid value~~), PRIORITY = PRIO
- 8. VERIFY Value_Source = VS
- 9. VERIFY Value_Source_Array = VS-~~SRC2~~, ARRAY_INDEX = PRIO
- ~~8. IF (Current_Command_Priority == PRIO) THEN~~
- ~~VERIFY Value_Source = SRC2~~
- 10. VERIFY Last_Command_Time = T1

7.3.1.28.5 Life Safety Value_Source Property Test

Reason for Change: Specify what is supposed to be written to the Value_Source property.

Purpose: To verify that the Value_Source property indicates the source of the current Mode property in a life safety object.

Test Concept: Select a life safety object which supports the Value Source mechanism. Mode is written, and it is verified that Value_Source is updated appropriately. Value_Source is then written to verify that the last writer can update it.

Notes to Tester: The value of the network-number in the network address of the Value_Source property will be one of three values. It will be 0 if the TD and IUT are on the same network and the IUT does not know the network number. It will be the network number of the local network if the IUT knows the network number and the TD and IUT are on the same network. It will be the TD's network number if the TD is on a different network.

Test Steps:

- 1. WRITE Mode = V1
- 2. VERIFY Mode = V1
- 3. VERIFY Value_Source = (TD's device identifier or network address)
- 4. WRITE Value_Source = (VS(TD's device identifier, O1, any valid object identifier ~~any valid value~~, V2)
- 5. VERIFY Value_Source = VS ~~V2~~

7.3.1.28.X1 Value Source Initiated Locally Test

Reason for Change: No test for this functionality.

Purpose: To verify that a change to the Present_Value initiated by the local device results in updates to the Value Source properties. This change to the Present_Value could be from a Program object, Schedule object or some mechanism internal to the device.

Test Concept: The Present_Value is changed by a mechanism local to the device. The Value_Source is verified and, if the object is commandable, Last_Command_Time, Value_Source_Array, and Command_Time_Array properties are checked to verify that they have been updated appropriately.

Configuration Requirements: If commandable, the value selected for PRIO shall be numerically small enough to be the active priority.

Test Steps:

1. MAKE (O1, Present_Value change by some mechanism local to the device. If the object is commandable, at a PRIORITY = PRIO)
2. IF (the mechanism is from an object, O2) THEN
3. VERIFY Value_Source = VS, (IUT's device identifier, O2)
4. ELSE
5. VERIFY Value_Source = VS, (IUT's device identifier)
6. IF (O1 is commandable) THEN
7. VERIFY Value_Source_Array = VS, ARRAY_INDEX = PRIO
8. VERIFY Last_Command_Time ~= (T1, the current local time)
9. IF (Command_Time_Array is present) THEN
10. VERIFY Command_Time_Array = T1, ARRAY_INDEX = PRIO

7.3.1.X73 Number_Of_States Property Tests

7.3.1.X73.1 Writable Number_Of_States Test

Reason for Change: New Test

Purpose: This test verifies that when the value of the Number_Of_States property is written, the size of the State_Text array is changed accordingly.

Test Concept: N1 and N2 are valid values of the Number_Of_States property, N1 and N2 do not equal the value of the Number_Of_States, and N1 does not equal N2. The value of the Number_Of_States property is written to N1 and the size of the State_Text and the value of the Number_Of_States property is verified. The procedure is repeated with N2 and again with N1.

Configuration Requirements: The IUT shall be configured with a Multi-state object O1, with a writable Number_Of_States property.

Test Steps:

1. WRITE O1, Number_Of_States = N1
2. VERIFY Number_Of_States = N1
3. VERIFY State_Text = N1, ARRAY INDEX = 0
4. WRITE O1, Number_Of_States = N2
5. VERIFY Number_Of_States = N2
6. VERIFY State_Text = N2, ARRAY INDEX = 0
7. WRITE O1, Number_Of_States = N1
8. VERIFY Number_Of_States = N1
9. VERIFY State_Text = N1, ARRAY INDEX = 0

7.3.2 Object Specific Tests

7.3.2.21 Notification Class Object

7.3.2.21.3 Recipient_List Tests

7.3.2.21.3.1 ValidDays Test

Reason For Change: Fix delay reference.

Purpose: To verify the operation of the Valid Days parameter of a BACnetDestination as used in the Recipient_List property of the Notification Class object.

Test Concept: The TD will select one instance of the Notification Class object and one instance of an event-generating object that is linked to the Notification Class object. The Recipient_List of the Notification Class object shall contain a single recipient with the Valid Days parameter configured so that at least one day is TRUE and at least one day is FALSE. The properties of the event-generating object will be manipulated to cause the Event_State to change from NORMAL to OFFNORMAL. The tester verifies that if the local date is one of the valid days a notification message is transmitted and if the local date is not a valid day then no notification message is transmitted. For devices that implement a read-only Recipient_List property for all instances of Notification Class objects and are exclusively configured for all days (Valid Days set to all Days), this test shall be omitted. For devices that implement a writeable Recipient_List property for all instances of Notification Class objects, and exclusively accept all days as the only permitted configuration, this test shall be omitted.

Configuration Requirements: The IUT shall be configured with one or more instance of the Notification Class object and at least one event-generating object that is linked to the Notification Class object. The event-generating object may be any object that supports intrinsic reporting or it may be an Event Enrollment object. The event-generating object shall have the Event_Enable property configured to transmit notification messages for all event transitions. The event-generating object shall be configured to be in a NORMAL event state at the start of the test. The Notification Class object shall be configured with a single recipient in the Recipient_List. The Valid Days parameter shall be configured so that at least one day of the week has a value of TRUE and at least one day of the week has a value of FALSE. The Transitions parameter shall be configured for the recipient to receive notifications for all event transitions.

Test Steps:

1. (TRANSMIT TimeSynchronization-Request,
 'Time' = (any time within the window defined by From Time and To Time in the BACnetDestination that
 corresponds to one of the valid days)) |
 (TRANSMIT UTCTimeSynchronization-Request,
 'Time' = (any time within the window defined by From Time and To Time in the BACnetDestination that
 corresponds to one of the valid days, converted to UTC)) |
 MAKE (the local date and time = (any time within the window defined by From Time and To Time in the
 BACnetDestination that corresponds to one of the valid days))
2. ~~WAIT (pTimeDelay + Notification Fail Time)~~ **WAIT (Internal Processing Fail Time)**
3. VERIFY pCurrentState = NORMAL
4. IF (pMonitoredValue is writable) THEN
 WRITE pMonitoredValue = (a value that is OFFNORMAL)
 ELSE
 MAKE (pMonitoredValue have a value that is OFFNORMAL)
5. WAIT (pTimeDelay)
6. BEFORE **Notification Fail Time**
 RECEIVE ConfirmedEventNotification-Request,
 'Process Identifier' = (any valid process ID),
 'Initiating Device Identifier' = IUT,
 'Event Object Identifier' = (the event-generating object configured for this test),
 'Time Stamp' = (any valid time stamp),

- 'Notification Class' = (the class corresponding to the object being tested),
- 'Priority' = (the value configured to correspond to a TO-OFFNORMAL transition),
- 'Event Type' = (any valid event type),
- 'Message Text' = (optional, any valid message text),
- 'Notify Type' = EVENT | ALARM,
- 'AckRequired' = TRUE | FALSE,
- 'From State' = NORMAL,
- 'To State' = OFFNORMAL,
- 'Event Values' = (values appropriate to the event type)
- 7. TRANSMIT BACnet-SimpleACK-PDU
- 8. VERIFY pCurrentState = OFFNORMAL
- 9. (TRANSMIT TimeSynchronization-Request,
 - 'Time' = (any time within the window defined by From Time and To time in the BACnetDestination that corresponds to one of the invalid days)) |
 - (TRANSMIT UTCTimeSynchronization-Request,
 - Time' = (any time within the window defined by From Time and To Time in the BACnetDestination that corresponds to one of the invalid days, converted to UTC)) |
 - MAKE (the local date and time = (any time within the window defined by From Time and To Time in the BACnetDestination that corresponds to one of the invalid days))
- 10. IF (pMonitoredValue is writable) THEN
 - WRITE pMonitoredValue = (a value that is NORMAL)
 - ELSE
 - MAKE (pMonitoredValue have a value that is NORMAL)
- 11. WAIT (~~pTimeDelayNormal~~~~pTimeDelay~~ + **Notification Fail Time**)
- 12. CHECK (verify that no notification message was transmitted)

7.3.2.23 Schedule Object Tests

7.3.2.23.8 List_Of_Object_Property_Reference External Test

Reason for Change: Test changed to match the test concept.

Purpose: To verify that the Schedule object writes to objects and properties contained within the IUT.

Test Concept: The Schedule object is configured to write to a property of another object within the same device. The IUT's clock is then set to a time between a pair of scheduled write operations, and verification of the first write operation's data value is performed. The time is advanced to the second time, the Schedule object's Present_Value is checked, and verifications of the write operations are performed. If the IUT does not support writing to object properties within the IUT, then this test shall not be performed.

Configuration Requirements: The IUT is configured with a Schedule object containing a List_Of_Object_Property_References property that references, if possible, at least one property in another object within the IUT. The Schedule object is configured with either a Weekly_Schedule or an active Exception_Schedule, during a period where Effective_Period is active, with at least two consecutive entries with distinguishable values in the List of BACnetTimeValues, and with no Exception_Schedules at a higher priority. D₁ represents the date and time of the first of these two BACnetTimeValues, with corresponding value V₁, while D₂ and V₂ (a value distinguishable from V₁) represent the second BACnetTimeValue. A time D_t is defined to occur between D₁ and D₂.

Test Steps:

1. (TRANSMIT TimeSynchronization-Request, 'Time' = D_t) |
 (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D_t) |
 MAKE (the local date and time = D_t)
2. WAIT **Schedule Evaluation Fail Time**
3. VERIFY Present_Value = V₁

4. VERIFY (value of referenced property in IUT) = V_1
- ~~5. BEFORE Schedule Evaluation Fail Time~~
- ~~6. RECEIVE WriteProperty Request,~~
~~'Object Identifier' = (the referenced object in the TD),~~
~~'Property Identifier' = (the referenced property in the TD),~~
~~'Property Value' = V_2 ,~~
~~'Priority' = (the value of the Schedule object's Priority_For_Writing property)~~
- ~~7. TRANSMIT BACnet SimpleACK PDU~~
5. (TRANSMIT TimeSynchronization-Request, 'Time' = D_2) |
 (TRANSMIT UTCTimeSynchronization-Request, 'Time' = D_2) |
 MAKE (the local date and time = D_2)
6. WAIT **Schedule Evaluation Fail Time**
7. VERIFY Present_Value = V_2
8. VERIFY (value of referenced property in IUT) = V_2
98. VERIFY Present_Value = C_1

7.3.2.25 Event Log Tests

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. *EVI* = 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)

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)

7.3.2.30 Notification Forwarder Object Tests

7.3.2.30.13 Persistence Tests

7.3.2.30.13.1 Recipient_List Persistence Test

Reason for Change: Moved closing curly bracket from after step 4 to after step 7.

Purpose: This test insures that Recipient_List property value is maintained through a device "restart".

Test Concept: Initialize the Recipient_List property with a known value, then cycle power or restart the IUT and verify the Recipient_List property value is maintained.

Configuration Requirements: Base setup 1 for Notification Forwarder object tests.

Test Steps:

1. MAKE (Recipient_List = {
 (all), -- Valid Days
 (all), -- From Time, To Time
 DEST_OBJ_ID, -- Recipient DI
 DEST_PROCESS_ID, -- Any Process Identifier
 DEST_CONF_NOTIF, -- Any Issue Confirmed Notifications
 {T, T, T} -- Transitions
 }) -- One list element
2. IF (IUT supports the ReinitializeDevice service) THEN {
3. TRANSMIT ReinitializeDevice-Request,
 'Reinitialized State of Device' = WARMSTART,
 'Password' = (any valid password) --if required by IUT
4. RECEIVE BACnet-SimpleACK-PDU
- ~~53.~~ CHECK (Did the IUT perform a WARMSTART restart)
- ~~64.~~ WAIT for restart to complete

```

75.    VERIFY Recipient_List = {(all),           -- Valid Days
                                (all),          -- From Time, To Time
                                DEST_OBJ_ID,     -- Recipient D1
                                DEST_PROCESS_ID, -- Process Identifier
                                DEST_CONF_NOTIF, -- Issue Confirmed Notifications
                                {T, T, T}       -- Transitions
                                }               -- One list element
    }
8-6. MAKE (The IUT reset by cycling power)
9-7. WAIT for restart to complete
10-8. VERIFY Recipient_List = {(all),           -- Valid Days
                                (all),          -- From Time, To Time
                                DEST_OBJ_ID,     -- Recipient D1
                                DEST_PROCESS_ID, -- Process Identifier
                                DEST_CONF_NOTIF, -- Issue Confirmed Notifications
                                {T, T, T}       -- Transitions
                                }               -- One list element

```

7.3.2.30.13.2 Subscribed_Recipients Persistence Test

Reason for Change: Move closing curly bracket from after step 3 to after step 6..

Purpose: This test insures that Subscribed_Recipients property values are maintained through a device “restart”.

Test Concept: Initialize the Subscribed_Recipients property with a known value, then cycle power to restart the IUT and verify the Subscribed_Recipients property value is maintained.

Configuration Requirements: Base setup 2 for Notification Forwarder object tests with TR lifetime sufficient for this test.

Note To Tester: Start_Up_TimeA or Start_Up_TimeB is the time in minutes required to restart the IUT.

Test Steps:

```

1.  IF (IUT supports the ReinitializeDevice service) THEN {
2.      TRANSMIT ReinitializeDevice-Request,
          'Reinitialized State of Device' = WARMSTART,
          'Password' = (any valid password) --if required by IUT
3.      RECEIVE BACnet-SimpleACK-PDU
    }
4-2. CHECK (Did the IUT perform a WARMSTART restart)
5-3. WAIT for restart to complete, making a note of the time to restart as Start_Up_TimeA
6-4. VERIFY Subscribed_Recipients =
    {DEST_OBJ_ID, -- Recipient D1
      DEST_PROCESS_ID, -- Process Identifier
      DEST_CONF_NOTIF, -- Issue Confirmed Notifications
      TR1 -- Time Remaining where (TR-Start_Up_TimeA-1) <= TR1 <= TR
    } -- One list element
    }
7-5. MAKE (The IUT reset by cycling power)
8-6. WAIT for restart to complete, making a note of the time to restart as Start_Up_TimeB
9-7. VERIFY Subscribed_Recipients =
    {DEST_OBJ_ID, -- Recipient D1
      DEST_PROCESS_ID, -- Process Identifier
      DEST_CONF_NOTIF, -- Issue Confirmed Notifications

```

```

TR1    -- Time Remaining where (TR-Start_Up_TimeA-Start_Up_TimeB-2) <= TR1 <= TR
}      -- One list element
    
```

~~Note To Tester: Start_Up_TimeA or Start_Up_TimeB is the time in minutes required to restart the IUT.~~

7.3.2.39 Lighting Output Object Tests

7.3.2.39.5 Lighting Command Operation RAMP_TO Test

Reason for change: wrong expected value for Tracking_Value in Step 25

Purpose: To verify the correct operation of RAMP_TO lighting command by observing the value of Present_Value, In_Progress and Tracking_Value.

Test Concept: The TD writes to Present_Value at each end of the range (i.e., 0% or 100%), and then writes to the Lighting Command Operation with RAMP_TO with a slow enough ramp rate to allow In_Progress and Tracking_Value to be observed while set to RAMP_ACTIVE. The Tracking_Value will be checked at the end of the ramp to verify that it tracked the target level. The IUT shall be tested for ramp up (0% to 100%) and ramp down (100% to 0%).

Configuration Requirements: O1 shall be configured such that all slots in the Priority_Array numerically less than PTY1 have a value of NULL and no internal algorithms are issuing commands to O1 at a priority numerically less than or equal to PTY1. V1 > 1 and V2 < 100%

Test Steps:

```

-- Start with 0% Present_Value to test ramp up
1. WRITE Present_Value = 0, ARRAY INDEX = PTY1
2. VERIFY Present_Value = 0
3. WAIT Internal Processing Fail Time
4. VERIFY Tracking_Value = 0

-- Write a RAMP_TO command (operation, target-value, priority, ramp-rate)
5. WRITE Lighting_Command = (RAMP_TO, V1, PTY1, any valid rate)
6. WAIT Internal Processing Fail Time
7. VERIFY Priority_Array = V1, ARRAY INDEX = PTY1
8. VERIFY Present_Value = V1

-- Check In_Progress while ramping up
9. VERIFY In_Progress = RAMP_ACTIVE

-- Make sure that Tracking_Value increases with the ramp-rate
10. WHILE (In_Progress <> IDLE) DO {
11. VERIFY Tracking_Value > 0 < V1
12. CHECK (Tracking_Value is increasing with the ramp-rate)}

-- When ramping up is completed, check In_Progress and Tracking_Value
13. VERIFY In_Progress = IDLE
14. VERIFY Tracking_Value = V1

-- Now repeat the test with 100% Present_Value to test ramp down
15. WRITE Present_Value = 100, ARRAY INDEX = PTY1
16. VERIFY Present_Value = 100
17. WAIT Internal Processing Fail Time
    
```

18. VERIFY Tracking_Value = 100

-- Write a RAMP_TO command (operation, target-value, priority, ramp-rate)

19. WRITE Lighting_Command = (RAMP_TO, V2, PTY1, any valid rate)

20. WAIT Internal Processing Fail Time

21. VERIFY Priority_Array = V2, ARRAY INDEX = PTY1

22. VERIFY Present_Value = V2

-- Check In_Progress while ramping up

23. VERIFY In_Progress = RAMP_ACTIVE,

-- Make sure that Tracking_Value decreases with the ramp-rate

24. WHILE (In_Progress <> RAMP_ACTIVE) DO {

25. VERIFY Tracking_Value < 100

26. VERIFY Tracking_Value > V2

27. CHECK (Tracking_Value is decreasing with the ramp-rate)}

-- Check In_Progress and Tracking_Value

28. VERIFY In_Progress = IDLE

29. VERIFY Tracking_Value = V2

7.3.2.41 Access Point Object Tests

7.3.2.41.4 Authorization Mode Test

Reason for Change: Remove testing for NONE Authorization_Mode.

Purpose: To verify each authorization mode supported by this IUT.

Test Concept: For each authorization mode supported by the IUT, a valid credential is presented at the access point to verify that the appropriate action is taken.

Configuration Requirements: See Clause 7.3.2.41. This test requires the following additional configuration:

- a) An active credential with valid access rights for the access point shall be represented by Access Credential object C1.
- b) An active credential with no valid access rights shall be represented by Access Credential object C2.

Note: If the VERIFICATION_REQUIRED or AUTHORIZATION_DELAYED mode is supported, the vendor must provide a mechanism for external verification to be performed.

Test Steps:

-- verify GRANT_ACTIVE mode

1. IF (GRANT_ACTIVE is supported) THEN
 - READ EventTag = Access_Event_Tag
 - WRITE Authorization_Mode = GRANT_ACTIVE
 - MAKE (present credential C2 at credential reader for this access point)
 - VERIFY Access_Event = GRANTED
 - VERIFY Access_Event_Time = (the time that credential C2 was presented)
 - VERIFY Access_Event_Credential = C2
 - VERIFY Access_Event_Tag = EventTag + 1

-- verify DENY_ALL mode

2. IF (DENY_ALL is supported) THEN
 - READ EventTag = Access_Event_Tag

```

WRITE Authorization_Mode = DENY_ALL
MAKE (present credential C1 at credential reader for this access point)
VERIFY Access_Event = DENIED_DENY_ALL
VERIFY Access_Event_Time = (the time that credential C1 was presented)
VERIFY Access_Event_Credential = C1
VERIFY Access_Event_Tag = EventTag + 1

```

-- verify VERIFICATION_REQUIRED mode (verification authorized)

```

3. IF (VERIFICATION_REQUIRED is supported) THEN
    READ EventTag = Access_Event_Tag
    WRITE Authorization_Mode = VERIFICATION_REQUIRED
    MAKE (present credential C1 at credential reader for this access point)
    VERIFY Access_Event = VERIFICATION_REQUIRED
    VERIFY Access_Event_Time = (the time that credential C1 was presented most recently)
    VERIFY Access_Event_Credential = C1
    VERIFY Authentication_Status = WAITING_FOR_VERIFICATION
    MAKE (external verification process grants access)
    VERIFY Access_Event = GRANTED
    VERIFY Access_Event_Time = (the time that verification process granted access)
    VERIFY Access_Event_Credential = C1
    VERIFY Access_Event_Tag = EventTag + 1

```

-- verify VERIFICATION_REQUIRED mode (verification denied)

```

    READ EventTag = Access_Event_Tag
    WRITE Authorization_Mode = VERIFICATION_REQUIRED
    MAKE (present credential C1 at credential reader for this access point)
    VERIFY Access_Event = VERIFICATION_REQUIRED
    VERIFY Access_Event_Time = (the time that credential C1 was presented)
    VERIFY Access_Event_Credential = C1
    VERIFY Authentication_Status = WAITING_FOR_VERIFICATION
    MAKE (external verification process denies access)
    VERIFY Access_Event = DENIED_VERIFICATION_FAILED
    VERIFY Access_Event_Time = (the time that verification process denied access)
    VERIFY Access_Event_Credential = C1
    VERIFY Access_Event_Tag + 1

```

-- verify VERIFICATION_REQUIRED mode (verification timeout)

```

    READ EventTag = Access_Event_Tag
    WRITE Authorization_Mode = VERIFICATION_REQUIRED
    MAKE (present credential C1 at credential reader for this access point)
    VERIFY Access_Event = VERIFICATION_REQUIRED
    VERIFY Access_Event_Time = (the time that credential C1 was presented)
    VERIFY Access_Event_Credential = C1
    VERIFY Authentication_Status = WAITING_FOR_VERIFICATION
    MAKE (external verification process does not respond within verification time)
    WAIT Verification_Time
    VERIFY Access_Event = DENIED_VERIFICATION_TIMEOUT
    VERIFY Access_Event_Time = (the time that verification process timed out)
    VERIFY Access_Event_Credential = C1
    VERIFY Access_Event_Tag = EventTag + 1

```

-- verify AUTHORIZATION_DELAYED mode (access granted)

```

4. IF (AUTHORIZATION_DELAYED is supported) THEN
    WRITE Authorization_Mode = AUTHORIZATION_DELAYED
    MAKE (present credential C1 at credential reader for this access point)
    VERIFY Access_Event = AUTHORIZATION_DELAYED

```



```

VERIFY Access_Event_Time = (the time that credential C1 was presented)
VERIFY Access_Event_Credential = C1
MAKE (external verification process does not respond within verification time)
WAIT Verification_Time
VERIFY Access_Event = GRANTED
VERIFY Access_Event_Time = (the time that verification process timed out)
VERIFY Access_Event_Credential = C1

```

```

-- verify AUTHORIZATION_DELAYED mode (access denied)
WRITE Authorization_Mode = AUTHORIZATION_DELAYED
READ EventTag = Access_Event_Tag
MAKE (present credential C1 at credential reader for this access point)
VERIFY Access_Event = AUTHORIZATION_DELAYED
VERIFY Access_Event_Time = (the time that credential C1 was presented)
VERIFY Access_Event_Credential = C1
MAKE (external verification process denies access)
VERIFY Access_Event = DENIED_VERIFICATION_FAILED
VERIFY Access_Event_Time = (the time that verification process denied access)
VERIFY Access_Event_Credential = C1
VERIFY Access_Event_Tag = EventTag + 1

```

-- verify NONE mode - *no tests to perform*

```

5. IF (NONE is supported) THEN
WRITE Authorization_Mode = NONE
WAIT Internal Processing Fail Time
VERIFY Authentication_Status = DISABLED

```

7.3.2.46 Network Port Object Tests

7.3.2.46.3 Network Port Command Tests

7.3.2.46.3.2 DISCARD_CHANGES Command Tests

7.3.2.46.3.2.2 DISCARD_CHANGES Command Failure Test

Reason for change: Incorrect error code.

Purpose: To verify that Network Port object responds to DISCARD_CHANGES commands when the command is not supported.

Test Concept: Attempt to command a Network Port which does not support the DISCARD_CHANGES. Verify that the attempt fails with an Error Class of PROPERTY and an error code of ~~VALUE_OUT_OF_RANGE~~ *OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED*.

Configuration Requirements: Select a Network Port which supports writable properties that set the Changes_Pending property to TRUE.

Test Steps:

1. TRANSMIT WriteProperty-Request,
'Object Identifier' = (the Network Port object),
'Property Identifier' = (any writable property that results in Changes_Pending = TRUE),
'Property Value' = (any valid value)

2. RECEIVE BACnet-SimpleACK-PDU
3. TRANSMIT WriteProperty-Request,
 'Object Identifier' = (the Network Port object),
 'Property Identifier' = Command,
 'Property Value' = DISCARD_CHANGES,
4. RECEIVE BACnet-Error-PDU
 'Error Class' = PROPERTY,
 'Error Code' = ~~VALUE_OUT_OF_RANGE~~ *OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED*
5. VERIFY Command = IDLE

7.3.2.46.3.6 RESTART_AUTONEGOTIATION Command Tests

7.3.2.46.3.6.2 RESTART_AUTONEGOTIATION Command Failure Test

Reason for Change: Updated to make the Link_Speed_Autonegotiate check optional.

Purpose: To verify that Network Port objects respond to the RESTART_AUTONEGOTIATION command with the correct error codes when the command is not supported / enabled.

Test Concept: Starting with a Network Port object which is not configured to auto-negotiate its link speed or which does not support the RESTART_AUTONEGOTIATION, command it to restart auto-negotiation. Verify that the correct error code is returned.

Configuration Requirements: If the network port support auto-negotiation, disable it. If the IUT does not support the Command property, or all Network Port object support auto-negotiation and it cannot be disabled, then this test shall be skipped.

Test Steps:

-- make sure our initial conditions are good

1. *IF Link_Speed_Autonegotiate is present THEN*
~~VERIFY Link_Speed_Auto_negotiate = TRUE~~
2. *VERIFY Link_Speed_Autonegotiate = TRUE*

-- request the renewal, and wait for it to timeout

23. TRANSMIT WriteProperty-Request,
 'Object Identifier' = (the Network Port object),
 'Property Identifier' = Command,
 'Property Value' = RESTART_AUTONEGOTIATION
34. *IF the port does not support auto-negotiation THEN*
5. RECEIVE BACnet-Error-PDU
 'Error Class' = PROPERTY,
 'Error Code' = OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED
- ELSE
6. RECEIVE BACnet-Error-PDU
 'Error Class' = PROPERTY,
 'Error Code' = VALUE_OUT_OF_RANGE

7.3.2.46.4 Hierarchical Network Port Tests

[Remove test 7.3.2.46.4.1]

7.3.2.46.X Valid Hierarchy Tests

7.3.2.46.X.1 Valid Hierarchy Chain - MSTP Test

Reason for Change: Explicit test for this data link.

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

Test Concept: Starting with a hierarchical Network Port object (NP_APP) 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.

Notes to Tester: NPOs of Network_Type = MSTP are excluded from this test if they are at Protocol_Level = PROTOCOL and they are not referenced by another NPO. NPOs are also excluded from this test if they are at Protocol_Level = PHYSICAL and Network_Type = SERIAL 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 = MSTP
 3. VERIFY NP_APP, Reference_Port <> 4194303
- Verify Protocol_Level = PROTOCOL
 4. RP = READ (NP_APP, Reference_Port)
 5. NP_PROT = (Network Port, RP)
 6. VERIFY NP_PROT, Protocol_Level = PROTOCOL
 7. VERIFY NP_PROT, Network_Type = MSTP
 8. VERIFY NP_PROT, Reference_Port <> 4194303
- Verify Protocol_Level = PHYSICAL
 9. RP = READ (NP_PROT, Reference_Port)
 10. NP_PHY = (Network Port, RP)
 11. VERIFY NP_PHY, Protocol_Level = PHYSICAL
 12. VERIFY NP_PHY, Network_Type = SERIAL
 13. VERIFY NP_PHY, Reference_Port = 4194303

7.3.2.46.X.2 Valid Hierarchy Chain - IPvX Test

Reason for Change: Explicit test for this data link.

Purpose: To verify that a hierarchical chain of IPV4 or IPV6 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 (NT) of IPV4 or IPV6 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.

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 are also excluded from this test if they are at Protocol_Level = PHYSICAL and Network_Type = ETHERNET 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 = NT
3.  RP = READ NP_APP, Reference_Port
4.  VERIFY RP <> 4194303

-- Verify required NPO at Protocol_Level = PROTOCOL
5.  NP_PROT = (Network Port, RP)
6.  VERIFY NP_PROT, Protocol_Level = PROTOCOL
7.  VERIFY NP_PROT, Network_Type = NT
8.  RP = READ NP_PROT, Reference_Port
9.  IF (RP <> 4194303) THEN
    {
      -- Check if Protocol_Level = PHYSICAL
10.   NPx = (Network Port, RP)
11.   IF (NPx, Protocol_Level = PHYSICAL) THEN
       -- PHYSICAL level. Could be ETHERNET or proprietary
12.     VERIFY NPx, Reference_Port = 4194303
    ELSE
      {
        -- Some number of NPOs at PROTOCOL level
        -- move down the chain of NPOs to the bottom
13.     WHILE (RP <> 4194303)
        {
14.       NPx = (Network Port, RP)
15.       VERIFY NPx, Protocol_Level <> (BACNET_APPLICATION OR NON_BACNET_APPLICATION)
          -- check that if PHYSICAL level it is the last NPO
16.       IF (NPx, Protocol_Level == PHYSICAL) THEN
17.         VERIFY NPx, Reference_Port = 4194303
18.         RP = READ NPx, Reference_Port
        }
        -- either the NPO is at PROTOCOL and does not reference another NPO or it is PHYSICAL
        -- If PHYSICAL, it could be ETHERNET or proprietary or SERIAL -- done
      }
    }
  ELSE
    {
      -- NPO is at PROTOCOL and does not reference another NPO - done
    }
  }
```

7.3.2.46.X.3 Valid Hierarchy Chain - Application and Physical Test

Reason for Change: Explicit test for data links without PROTOCOL level NPOs.

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

Test Concept: Starting with a hierarchical Network Port object (NP_APP) which represents a configured application layer port (BACNET_APPLICATION), verify this port contains the correct Network_Type (NT), Protocol_Level, and Reference_Port. Visit the next level ensuring that the Network_Type, Protocol_Level, and Reference_Port are correct.

Notes to Tester: NPOs of this Network_Type with Protocol_Level = PHYSICAL are excluded from this test if 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 = NT
3.  VERIFY NP_APP, Reference_Port <> 4194303

-- Verify Protocol_Level = PHYSICAL
4.  RP = READ (NP_APP, Reference_Port)
5.  NP_PHY = (Network Port, RP)
6.  VERIFY NP_PHY, Protocol_Level = PHYSICAL
7.  VERIFY NP_PHY, Network_Type = NT
8.  VERIFY NP_PHY, Reference_Port = 4194303
```

7.3.2.46.X.4 Valid Hierarchy Chain - Secure Connect Test

Reason for Change: Explicit test for this data link.

Purpose: To verify that a hierarchical chain of SECURE_CONNECT 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 of SECURE_CONNECT 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.

Test Steps:

```
-- Verify Protocol_Level = BACNET_APPLICATION
1.  VERIFY NP_APP, Protocol_Level = BACNET_APPLICATION
2.  VERIFY NP_APP, Network_Type = SECURE_CONNECT
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)
8.      VERIFY NPx, Network_Type <> SECURE_CONNECT
      -- check that if PHYSICAL level it is the last NPO
9.      IF (NPx, Protocol_Level == PHYSICAL) THEN
10.         VERIFY NPx, Reference_Port = 4194303
11.         RP = READ NPx, Reference_Port
    }
    -- either the NPO is at PROTOCOL and does not reference another NPO or it is PHYSICAL

12. IF (NP_APP, Additional_Reference_Ports is present) THEN {
13.     REPEAT ARP = (for each entry in NP_APP, Additional_Reference_Ports) DO {
14.         NPx = (Network Port, ARP)
15.         VERIFY NPx, Protocol_Level <> (NON_BACNET_APPLICATION or BACNET_APPLICATION)
16.         VERIFY NPx, Network_Type <> SECURE_CONNECT
```

```

-- check that if PHYSICAL level it is the last NPO
17. IF (NPx, Protocol_Level == PHYSICAL) THEN
18.     IF (NPx, Additional_Reference_Ports is present) THEN
19.         VERIFY NPx, Additional_Reference_Ports = (empty list)
-- Check the rest of the chain
20. RP = READ NPx, Reference_Port
21. WHILE (RP <> 4194303)
    {
22.     NPx = (Network Port, RP)
23.     VERIFY NPx, Protocol_Level <> (NON_BACNET_APPLICATION
        or BACNET_APPLICATION)
24.     VERIFY NPx, Network_Type <> SECURE_CONNECT
-- check that if PHYSICAL level it is the last NPO
25.     IF (NPx, Protocol_Level == PHYSICAL) THEN
26.         VERIFY NPx, Reference_Port = 4194303
27.     RP = READ NPx, Reference_Port
    }
    }
}

```

7.3.2.46.X.5 Valid Hierarchy Chain - Proprietary Test

Reason for Change: Explicit test for this data link.

Purpose: To verify that a hierarchical chain of non-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 (NT) which represents a configured application layer port (NON_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 = NON_BACNET_APPLICATION
1.  VERIFY NP_APP, Protocol_Level = NON_BACNET_APPLICATION
2.  VERIFY NP_APP, Network_Type = NT
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.             {
21.                 NPx = (Network Port, RP)
22.                 VERIFY NPx, Protocol_Level <> (NON_BACNET_APPLICATION
                    or BACNET_APPLICATION)
                -- check that if PHYSICAL level it is the last NPO
23.                 IF (NPx, Protocol_Level == PHYSICAL) THEN
24.                     VERIFY NPx, Reference_Port = 4194303
25.                     RP = READ NPx, Reference_Port
26.             }
27.         }
28.     }
29. }

```

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
    }
    }
    }

```

7.3.2.47 Timer Object

7.3.2.47.1 Positive Tests

7.3.2.47.1.12 Forcing Timer Expiration by Writing IDLE

Reason for Change: In Step 7 verifying the exact Update_Time against the current date and time is not possible.

Purpose: Interrupting the Timer while it is RUNNING, via a value of IDLE written to the Timer_State property.

Test Concept: Configure and start the Timer T1 to operate according to its values. Then write IDLE to Timer_State and observe that specified properties take their required values and all configured State_Change_Values transitions if any, take place.

Configuration Requirements: T1 starts this test with the Timer_State equal to RUNNING.

Test Steps:

1. VERIFY Timer_Running = TRUE
2. VERIFY Timer_State = RUNNING
3. WRITE Timer_State = IDLE
4. CHECK (IUT exhibits any changes configured in RUNNING_TO_IDLE transition)
5. VERIFY Timer_State = IDLE

6. VERIFY Last_State_Change = RUNNING_TO_IDLE
7. IF (Expiration_Time property is present in T1) THEN
8. VERIFY Expiration_Time = (the unspecified datetime value)
89. IF (Update_Time property is present in T1) THEN
- ~~VERIFY Update_Time = (the current date and time)~~
10. VERIFY Update_Time ~= (the current date and time)
911. VERIFY Present_Value = 0

7.3.2.50 Staging Object Tests

7.3.2.50.11 Out_Of_Service, Status_Flags, and Reliability for Staging Object

Reason for Change: CONFIGURATION_ERROR added in step 13, it was previously omitted.

Purpose: To verify that Present_Value and Reliability are writable when Out_Of_Service is TRUE, to verify the relationship between Out_Of_Service, Status_Flags, and Reliability, and to verify that writes to Target_References only occur when Out_Of_Service is FALSE.

Test Concept: The Out_Of_Service property is set to TRUE and the value of the Status_Flags property is validated. Present_Value is modified to verify that Present_Stage evaluates but writes to Target_References do not occur. If the IUT supports Reliability values other than NO_FAULT_DETECTED, writability for that property is tested and the value of the Status_Flags property is validated. The Out_Of_Service property is set to FALSE and the value of the Status_Flags property is validated. The Present_Value for one of the Target_References is checked to verify that it has the correct value, indicative of a write that occurred when transitioning Out_Of_Service from TRUE to FALSE.

Configuration Requirements: The Staging object used for this test shall be configured with at least one object in the Target_References property. The Stages property shall be configured with two stages such that Stages[S].Values = {V1...} and Stages[S+1].Values = {V2...} where V1 <> V2. At the start of the test, the Staging object is properly configured such that Reliability = NO_FAULT_DETECTED and Present_Stage = S.

Test Steps:

1. READ SF1 = Status_Flags
2. VERIFY Reliability = NO_FAULT_DETECTED
3. VERIFY Present_Stage = S
4. READ O1 = Target_References, ARRAY INDEX = 1
5. VERIFY O1, Present_Value = V1
6. IF (Out_Of_Service is writable) THEN
- WRITE Out_Of_Service = TRUE
- ELSE
- MAKE (Out_Of_Service TRUE)
7. VERIFY Out_Of_Service = TRUE
8. VERIFY Status_Flags = (?, ?, TRUE)
9. WRITE Present_Value = (PV: (Stages[S].Limit + Stages[S].Deadband) < PV < Stages[S+1].Limit)
10. VERIFY Present_Value = PV
11. VERIFY Present_Stage = S+1
12. VERIFY O1, Present_Value = V1
13. IF (the IUT supports Reliability values other than NO_FAULT_DETECTED) THEN
- REPEAT X = (all values of the Reliability enumeration appropriate to the object type except
- NO_FAULT_DETECTED and CONFIGURATION_ERROR) DO {
- WRITE Reliability = X
- VERIFY Reliability = X
- VERIFY Status_Flags = (?, TRUE, ?, TRUE)
- WRITE Reliability = NO_FAULT_DETECTED
- VERIFY Reliability = NO_FAULT_DETECTED
- VERIFY Status_Flags = (?, FALSE, ?, TRUE)
- }

```

    }
14. IF (Out_Of_Service is writable) THEN
    WRITE Out_Of_Service = FALSE
    ELSE
    MAKE (Out_Of_Service FALSE)
15. VERIFY Status_Flags = SF1
16. VERIFY Reliability = NO_FAULT_DETECTED
17. IF (Present_Stage = S+1) THEN
    VERIFY O1, Present_Value = V2

```

8. APPLICATION SERVICE INITIATION TESTS

8.2 ConfirmedCOVNotification Service Initiation Tests

8.2.1 Change of Value Notification for Changes to Present_Value in Objects with a COV_Increment

Reason for Change: Defined value for ReportedPV in step 4 for later use.

Purpose: To verify that the IUT can initiate ConfirmedCOVNotification service requests conveying a change of the Present_Value property in Numeric Objects.

Test Concept: A subscription for COV notifications is established, using a Lifetime of L. L shall be set to a value less than 24 hours and large enough to complete the test. The Present_Value of the monitored object is changed by an amount less than the COV increment and it is verified that no COV notification is received. The Present_Value is then changed by an amount greater than the COV increment and a notification shall be received. The Present_Value may be changed using the WriteProperty service or by another means such as changing the input signal represented by an Analog Input object. For some implementations it may be necessary to write to the Out_Of_Service property first to accomplish this task. For implementations where it is not possible to write to these properties at all the vendor shall provide an alternative trigger mechanism to accomplish this task. All of these methods are equally acceptable.

Configuration Requirements: At the beginning of the test, the Out_Of_Service property shall have a value of FALSE. Select an object where Present_Value is not expected to change outside the tester's control by more than COV_Increment or which has a writable Out_Of_Service. In devices where the COV_Increment is always less than the minimal change that Present_Value can make, skip steps 8 through 10.

Notes to Tester: The IUT may initiate additional COVNotifications. The final COVNotification shall accurately reflect Present_Value and Status_Flags.

Test Steps:

```

REPEAT X = (one supported object of each type) DO {
1. TRANSMIT SubscribeCOV-Request,
    'Subscriber Process Identifier' = (any value > 0 chosen by the TD),
    'Monitored Object Identifier' = X,
    'Issue Confirmed Notifications' = TRUE,
    'Lifetime' = L
2. RECEIVE BACnet-SimpleACK-PDU
3. BEFORE Notification Fail Time
4. RECEIVE ConfirmedCOVNotification-Request,
    'Subscriber Process Identifier' = (the same value used in step 1),
    'Initiating Device Identifier' = IUT,
    'Monitored Object Identifier' = X,
    'Time Remaining' = (any value appropriate for the Lifetime selected),
    'List of Values' = (ReportedPV = the initial Present_Value and initial Status_Flags)
54. TRANSMIT BACnet-SimpleACK-PDU

```

```

65. TRANSMIT ReadProperty-Request,
    'Object Identifier' = X,
    'Property Identifier' = COV_Increment
76. RECEIVE BACnet-ComplexACK-PDU,
    'Object Identifier' = X,
    'Property Identifier' = COV_Increment,
    'Property Value' = (a value "increment" that will be used below)
87. IF (Out_Of_Service is writable) THEN
9.    WRITE X, Out_Of_Service = TRUE
10.   BEFORE Notification Fail Time
11.    RECEIVE ConfirmedCOVNotification-Request,
        'Subscriber Process Identifier' = (the same value used in step 1),
        'Initiating Device Identifier' = IUT,
        'Monitored Object Identifier' = X,
        'Time Remaining' = (any value appropriate for the Lifetime selected),
        'List of Values' = (ReportedPV = the current Present_Value, and new Status_Flags)
12.    TRANSMIT BACnet-SimpleACK-PDU
13. IF (Present_Value is now writable) THEN
14.    WRITE X, Present_Value = (any value that differs from ReportedPV by less than "increment")
    ELSE
15.    MAKE (Present_Value = any value that differs from ReportedPV by less than "increment")
16. WAIT Notification Fail Time
17. CHECK (verify that no COV notification was transmitted)
18. IF (Present_Value is now writable) THEN
19.    WRITE X, Present_Value = (any value that differs from ReportedPV by an amount greater than "increment")
    ELSE
20.    MAKE (Present_Value = any value that differs from ReportedPV by an amount greater than "increment")
21. BEFORE NotificationFailTime
22. RECEIVE ConfirmedCOVNotification-Request,
        'Subscriber Process Identifier' = (the same value used in step 1),
        'Initiating Device Identifier' = IUT,
        'Monitored Object Identifier' = X,
        'Time Remaining' = (any value appropriate for the Lifetime selected),
        'List of Values' = (the new Present_Value and new Status_Flags)
23. TRANSMIT BACnet-SimpleACK-PDU
24. TRANSMIT SubscribeCOV-Request,
        'Subscriber Process Identifier' = (the same value used in step 1),
        'Monitored Object Identifier' = X
25. RECEIVE BACnet-SimpleACK-PDU
26. IF (Out_Of_Service is writable) THEN
27.    WRITE X, Out_Of_Service = FALSE
}

```

8.2.19 Change of Value Notification from Other Standard Object Types (ConfirmedCOVNotification)

Reason for Change: Changed test name for clarity.

Purpose: To verify that the IUT can initiate ConfirmedCOVNotification service requests conveying a change of value for an object (O1) not listed in 135-2020 Table 13-1.

Test Concept: A subscription for COV notifications is established, using a Lifetime of L. L shall be set to a value less than 24 hours and large enough to complete the test. The value of the Present_Value is changed, and it is verified that a COV notification is received. If the Status_Flags is present and can be made to change, it is verified that when Status_Flags changes a COV notification is received.

Configuration Requirements: None.

Test Steps:

1. TRANSMIT SubscribeCOV-Request,
 'Subscriber Process Identifier' = (P1, any value \neq 0 chosen by the TD),
 'Monitored Object Identifier' = O1,
 'Issue Confirmed Notifications' = TRUE,
 'Lifetime' = L
2. RECEIVE BACnet-SimpleACK-PDU
3. BEFORE Notification Fail Time
4. RECEIVE ConfirmedCOVNotification-Request,
 'Subscriber Process Identifier' = (P1),
 'Initiating Device Identifier' = IUT,
 'Monitored Object Identifier' = O1,
 'Time Remaining' = (any value appropriate for the Lifetime selected),
 'List of Values' = (the initial values for Present_Value and Status_Flags (if O1 supports Status_Flags))
5. TRANSMIT BACnet-SimpleACK-PDU
6. MAKE (Present_Value change)
7. BEFORE Notification Fail Time
8. RECEIVE ConfirmedCOVNotification-Request,
 'Subscriber Process Identifier' = (P1),
 'Initiating Device Identifier' = IUT,
 'Monitored Object Identifier' = O1,
 'Time Remaining' = (any value appropriate for the Lifetime selected),
 'List of Values' = (updated values for Present_Value and Status_Flags (if O1 supports Status_Flags))
9. IF (Status_Flags are present and can be changed by some action) THEN {
10. MAKE (Status_Flags change)
11. BEFORE Notification Fail Time
12. RECEIVE ConfirmedCOVNotification-Request,
 'Subscriber Process Identifier' = (P1),
 'Initiating Device Identifier' = IUT,
 'Monitored Object Identifier' = O1,
 'Time Remaining' = (any value appropriate for the Lifetime selected),
 'List of Values' = (updated values for Present_Value, Status_Flags)
13. TRANSMIT BACnet-SimpleACK-PDU
- }

8.3 UnconfirmedCOVNotification Service Initiation Tests

8.3.21 - Change of Value Notification from Other Standard Object Types (UnconfirmedCOVNotification)

Reason for Change: Changed test name for clarity.

Purpose: To verify that the IUT can initiate UnconfirmedCOVNotification service requests conveying a change of value for an object (O1) not listed in 135-2020 Table 13-1.

Test Steps: The steps for this test case are identical to the test steps in 8.2.X2 except that the SubscribeCOV service request in step 1 shall have a value of FALSE for the 'Issue Confirmed Notifications' parameter, all of the ConfirmedCOVNotification requests shall be UnconfirmedCOVNotification requests, and there is no acknowledgment of the unconfirmed services. The MAC address used for the notification message shall be such that the TD is one of the recipients.

8.4 ConfirmedEventNotification Service Initiation Tests

8.4.14 UNSIGNED_RANGE Test (ConfirmedEventNotification Test)

[Note: Test renumbered without using standard change formatting]

Reason for change: wrong property name for high and low limit in Step 23 (Low_Limit and High_Limit instead of pLowLimit and pHighLimit.

Purpose: To verify the correct operation of the UNSIGNED_RANGE event algorithm.

Test Concept: This test is the same as 8.4.6, except that the Event_Type is UNSIGNED_RANGE instead of OUT_OF_RANGE, and there is no pDeadband. If pMonitoredValue is not under the tester's control in the IUT, then pHighLimit and/or pLowLimit are modified to generate event notifications. The object begins the test in a NORMAL state. pMonitoredValue is raised to a value that is above the high limit. After the time delay expires, the object should enter the HIGH_LIMIT state and transmit an event notification message. pMonitoredValue is lowered to a value that is below the high limit. After the time delay expires, the object should enter the NORMAL state and issue an event notification. The same process is repeated to test the low limit.

Configuration Requirements: If possible, the IUT shall be configured such that the Event_Enable property has a value of TRUE for the TO_OFFNORMAL and TO_NORMAL transitions. If possible, pLimitEnable shall have a value of TRUE for both HighLimit and LowLimit events. The 'Issue Confirmed Notifications' parameter in the Recipient_List of the configured Notification Class shall have a value of TRUE. The Recipient_List of the configured Notification Class shall contain the TD, thus ensuring that notifications are emitted. The event-generating objects shall be in a NORMAL state at the start of the test.

Notes to Tester: The time stamps indicated by "" can have a value that indicates an unspecified time or a time that precedes the timestamp of the first received notification.*

Test Steps:

1. VERIFY pCurrentState = NORMAL
2. IF (pMonitoredValue is writable) THEN
3. WRITE pMonitoredValue = (a value x: (x > pHighLimit))
- ELSE
4. MAKE (pMonitoredValue have a value x: (x > pHighLimit))
5. WAIT (pTimeDelay)
6. BEFORE **Notification Fail Time**
7. RECEIVE ConfirmedEventNotification-Request,
 - 'Process Identifier' = (any valid process ID),
 - 'Initiating Device Identifier' = IUT,
 - 'Event Object Identifier' = (the object being tested),
 - 'Time Stamp' = (Toffnormal: any valid time stamp),
 - 'Notification Class' = (the configured notification class),
 - 'Priority' = (the value configured to correspond to a TO_OFFNORMAL transition),
 - 'Event Type' = UNSIGNED_RANGE,
 - 'Message Text' = (optional, any valid message text),
 - 'Notify Type' = EVENT | ALARM,
 - 'AckRequired' = TRUE | FALSE,
 - 'From State' = NORMAL,
 - 'To State' = HIGH_LIMIT,
 - 'Event Values' = pMonitoredValue, pStatusFlags, pHighLimit
8. TRANSMIT BACnet-SimpleACK-PDU
9. IF (Protocol_Revision is present AND Protocol_Revision >= 13)) THEN
10. VERIFY Status_Flags = (TRUE, FALSE, ?, ?)
11. VERIFY pCurrentState = HIGH_LIMIT
12. IF (Protocol_Revision is present AND Protocol_Revision >= 1) THEN
13. VERIFY Event_Time_Stamps = (Toffnormal, *, *)

14. IF (pMonitoredValue is writable) THEN
15. WRITE pMonitoredValue = (a value x: (pLowLimit < x < pHighLimit))
16. ELSE
17. MAKE (pMonitoredValue have a value x: (pLowLimit < x < pHighLimit))
18. WAIT (pTimeDelayNormal)
19. BEFORE **Notification Fail Time**
20. RECEIVE ConfirmedEventNotification-Request,
 - 'Process Identifier' = (any valid process ID),
 - 'Initiating Device Identifier' = IUT,
 - 'Event Object Identifier' = (the object being tested),
 - 'Time Stamp' = (Tnormal: any valid time stamp),
 - 'Notification Class' = (the configured notification class),
 - 'Priority' = (the value configured to correspond to a TO_NORMAL transition),
 - 'Event Type' = UNSIGNED_RANGE,
 - 'Message Text' = (optional, any valid message text),
 - 'Notify Type' = EVENT | ALARM,
 - 'AckRequired' = TRUE | FALSE,
 - 'From State' = HIGH_LIMIT,
 - 'To State' = NORMAL,
 - 'Event Values' = pMonitoredValue, pStatusFlags, pHighLimit
21. TRANSMIT BACnet-SimpleACK-PDU
22. IF (Protocol_Revision is present AND Protocol_Revision >= 13)) THEN
23. VERIFY Status_Flags = (FALSE, FALSE, ?, ?)
24. VERIFY pCurrentState = NORMAL
25. IF (Protocol_Revision is present AND Protocol_Revision >= 1) THEN
26. VERIFY Event_Time_Stamps = (Toffnormal, *, Tnormal)
27. IF (pMonitoredValue is writable) THEN
28. WRITE pMonitoredValue = (a value x: (x < pLowLimit))
29. ELSE
30. MAKE (pMonitoredValue have a value x: (x < pLowLimit))
31. WAIT (pTimeDelay)
32. BEFORE **Notification Fail Time**
33. RECEIVE ConfirmedEventNotification-Request,
 - 'Process Identifier' = (any valid process ID),
 - 'Initiating Device Identifier' = IUT,
 - 'Event Object Identifier' = (the object being tested),
 - 'Time Stamp' = (Tlowlimit: any valid time stamp),
 - 'Notification Class' = (the configured notification class),
 - 'Priority' = (the value configured to correspond to a TO_OFFNORMAL transition),
 - 'Event Type' = UNSIGNED_RANGE,
 - 'Message Text' = (optional, any valid message text),
 - 'Notify Type' = EVENT | ALARM,
 - 'AckRequired' = TRUE | FALSE,
 - 'From State' = NORMAL,
 - 'To State' = LOW_LIMIT,
 - 'Event Values' = pMonitoredValue, pStatusFlags, pLowLimit
34. TRANSMIT BACnet-SimpleACK-PDU
35. IF (Protocol_Revision is present AND Protocol_Revision >= 13)) THEN
36. VERIFY Status_Flags = (TRUE, FALSE, ?, ?)
37. VERIFY pCurrentState = LOW_LIMIT
38. IF (Protocol_Revision is present AND Protocol_Revision >= 1) THEN
39. VERIFY Event_Time_Stamps = (Tlowlimit, *, Tnormal)
40. IF (pMonitoredValue is writable) THEN
41. WRITE pMonitoredValue = (a value x: (~~Low_Limit < x < High_Limit~~) (pLowLimit < x < pHighLimit))
42. ELSE
43. MAKE (pMonitoredValue have a value x: (~~Low_Limit < x < High_Limit~~) (pLowLimit < x < pHighLimit))

41. WAIT (pTimeDelayNormal)
42. BEFORE **Notification Fail Time**
43. RECEIVE ConfirmedEventNotification-Request,
 - 'Process Identifier' = (any valid process ID),
 - 'Initiating Device Identifier' = IUT,
 - 'Event Object Identifier' = (the object being tested),
 - 'Time Stamp' = (Tlowtonormal: any valid time stamp),
 - 'Notification Class' = (the configured notification class),
 - 'Priority' = (the value configured to correspond to a TO_NORMAL transition),
 - 'Event Type' = UNSIGNED_RANGE,
 - 'Message Text' = (optional, any valid message text),
 - 'Notify Type' = EVENT | ALARM,
 - 'AckRequired' = TRUE | FALSE,
 - 'From State' = LOW_LIMIT,
 - 'To State' = NORMAL,
 - 'Event Values' = pMonitoredValue, pStatusFlags, pLowLimit
44. TRANSMIT BACnet-SimpleACK-PDU
45. IF (Protocol_Revision is present AND Protocol_Revision >= 13)) THEN
46. VERIFY Status_Flags = (FALSE, FALSE, ?, ?)
47. VERIFY pCurrentState = NORMAL
48. IF (Protocol_Revision is present AND Protocol_Revision >= 1) THEN
49. VERIFY Event_Time_Stamped = (Tlowlimit, *, Tlowtonormal)

~~Notes to Tester: The time stamps indicated by "*" can have a value that indicates an unspecified time or a time that~~

8.7 GetEnrollmentSummary Service Initiation Tests

[Remove 135.1-2025, Clause 8.7 including all tests]

8.21 ReadRange Service Initiation Tests

8.21.9 Presents Log Records

Purpose: To verify that the IUT can initiate one or more ReadRange requests that access and present a tester-specified portion of log records. It is a generic test used to test data presentation requirements.

Test Concept: Run test in Clause 8.21.8 and verify that the data presentation meets the criteria specified by the BIBB being tested.

Notes to Tester: The values presented by the IUT may differ from the values transmitted on the wire due to rounding, truncation, formatting, language, conversion, etc.

~~Notes to Tester: The IUT is not required to display records containing log status values.~~

8.45 You-Are Service Initiation Tests

8.45.1 Configures Other Device's MAC Address

Reason for Change: xxxx.

Purpose: To verify the IUT can configure another device's MAC address using the You-Are service without relying on the Who-Am-I service.

Test Concept: The IUT configures TD with an appropriate MAC address without TD first sending a Who-Am-I.

Configuration Requirements: TD's Device object is configured with a known Device object instance number and no configured MAC address.

Notes to Tester: The IUT may require the tester to specify all the parameters needed to configure TD with You-Are, using the IUT's software. The destination address used by TD shall be selected such that the IUT will receive the messages.

Test Steps:

1. MAKE (the IUT configure TD)
2. RECEIVE
 - DESTINATION = LOCAL BROADCAST | GLOBAL BROADCAST | REMOTE BROADCAST,
 - You-Are Request,
 - 'Vendor Identifier' = (TD's Vendor_Identifier),
 - 'Model Name' = (TD's Model_Name),
 - 'Serial Number' = (TD's Serial_Number),
 - ~~'Device Identifier' = (TD's Device object),~~
 - 'Device MAC Address' = (an appropriate MAC address)
3. IF (TD is not an MS/TP subordinate node)
4. TRANSMIT
 - DESTINATION = IUT | LOCAL BROADCAST | GLOBAL BROADCAST | REMOTE BROADCAST,
 - I-Am Request,
 - 'Device Identifier' = (TD's Device object)
 - 'Max APDU Length Accepted' = (any valid value),
 - 'Segmentation Supported' = (any valid value),
 - 'Vendor Identifier' = (TD's Vendor_Identifier)

9. APPLICATION SERVICE EXECUTION TESTS

9.1 AcknowledgeAlarm Service Execution Tests

9.1.2 Negative AcknowledgeAlarm Service Execution Tests

9.1.2.1 Unsuccessful Alarm Acknowledgment of Confirmed Event Notifications Because the 'Time Stamp' is Too Old

Reason for Change: Changes required dur to 135-2016br-4 and deprecation of the 'time' form of the BACnetTimeStamp datatype.

Purpose: To verify that an alarm remains unacknowledged if the time stamp in the acknowledgment does not match the most recent transition to the current alarm state.

Test Concept: An alarm is triggered that causes the IUT to notify the TD and one other device. The TD acknowledges the alarm using an old time stamp and verifies that the acknowledgment is not accepted by the IUT and that the IUT does not notify other devices that the alarm was acknowledged. The TD then acknowledges the alarm using the proper time stamp and verifies that the acknowledgment is properly noted by the IUT. The IUT notifies all other recipients that the alarm was acknowledged.

Configuration Requirements: The IUT shall be configured with at least one object that can detect alarm conditions and send confirmed notifications. The Acked_Transitions property shall have the value B'111' indicating that all transitions have been acknowledged. The TD and one other BACnet device, if the IUT supports multiple recipients, shall be recipients of the alarm notification. D1 is either the pTimeDelay, or pTimeDelayNormal parameter, or 0 (for transitions to and from FAULT state) depending on the event transition.

Notes to Tester: The destination address used for the acknowledgment notification in step 11 shall be the same address used in step 3. If the IUT can only be configured with one recipient in the Recipient_List property of the issuing Notification_class object, omit steps 5, 6, 15, and 16.

Test Steps:

1. MAKE (a change that triggers the detection of an alarm event in the IUT)
2. WAIT (D1)
3. BEFORE **Notification Fail Time**
4. RECEIVE ConfirmedEventNotification-Request,
 - 'Process Identifier' = (PID: the process identifier configured for this event),
 - 'Initiating Device Identifier' = IUT,
 - 'Event Object Identifier' = (OI: the object detecting the alarm),
 - 'Time Stamp' = (T1: any valid time stamp),
 - 'Notification Class' = (NC: the notification class configured for this event),
 - 'Priority' = (PI: the priority configured for this event type),
 - 'Event Type' = (E1: any valid event type),
 - 'Message Text' = (MT: optional, any valid message text),
 - 'Notify Type' = (NT: the notify type configured for the event),
 - 'AckRequired' = TRUE,
 - 'From State' = (S1: any appropriate event state),
 - 'To State' = (S2: any appropriate event state),
 - 'Event Values' = (the values appropriate to the event type)
45. TRANSMIT BACnet-SimpleACK-PDU
56. RECEIVE
 - DESTINATION = (a device other than the TD),
 - SOURCE = IUT,
 - ConfirmedEventNotification-Request,
 - 'Process Identifier' = (PID: the process identifier configured for this event),
 - 'Initiating Device Identifier' = IUT,
 - 'Event Object Identifier' = (OI: the object detecting the alarm),
 - 'Time Stamp' = (T1),
 - 'Notification Class' = (NC: the notification class configured for this event),
 - 'Priority' = (PI: the priority configured for this event type),
 - 'Event Type' = (E1/E2: any valid event type),
 - 'Message Text' = (MT optional, any valid message text),
 - 'Notify Type' = (NT: the notify type configured for the event),
 - 'AckRequired' = TRUE,
 - 'From State' = (S1: any appropriate event state),
 - 'To State' = (S2: any appropriate event state),
 - 'Event Values' = (the values appropriate to the event type)
67. TRANSMIT BACnet-SimpleACK-PDU
78. VERIFY (the 'Event Object Identifier' from the event notification) OI,
 - Acked_Transitions = (appropriate bit FALSE, the others TRUE)
89. TRANSMIT AcknowledgeAlarm-Request,
 - 'Acknowledging Process Identifier' = (P1: the value of the 'Process Identifier' parameter in the event notification),
 - 'Event Object Identifier' = (OI: the 'Event Object Identifier' from the event notification),
 - 'Event State Acknowledged' = (S2: the state specified in the 'To State' parameter of the notification),
 - 'Time Stamp' = (any valid time stamp older than T1),
 - 'Acknowledgment Source' = (any valid value)
 - 'Time of Acknowledgment' = (the current time using a Time format)
910. RECEIVE BACnet-Error-PDU
 - Error Class = SERVICES,
 - Error Code = INVALID_TIME_STAMP
1011. VERIFY (the 'Event Object Identifier' from the event notification) OI,

Acked_Transitions = (appropriate bit FALSE, the others TRUE)

~~14/2.~~ TRANSMIT AcknowledgeAlarm-Request,
 'Acknowledging Process Identifier' = (~~PID~~the process identifier configured for this event),
 'Event Object Identifier' = (~~E1~~the 'Event Object Identifier' from the event notification),
 'Event State Acknowledged' = (~~S2~~the state specified in the 'To State' parameter of the notification),
 'Time Stamp' = T1,
 'Time of Acknowledgment' = (the current time using a Time format)

~~12/3.~~ RECEIVE BACnet-SimpleACK-PDU

~~13.~~ IF (Protocol_Revision is present AND Protocol_Revision >= 1) THEN

~~14.~~ BEFORE Notification Fail Time

~~14/5.~~ RECEIVE

ConfirmedEventNotification-Request,
 'Process Identifier' = (~~PID~~the process identifier configured for this event),
 'Initiating Device Identifier' = IUT,
 'Event Object Identifier' = (~~O1~~the object detecting the alarm),
 'Time Stamp' = (T2: any valid time stamp),
 'Notification Class' = (~~NC~~the notification class configured for this event),
 'Priority' = (~~P1~~the priority configured for this event type),
 'Event Type' = (~~E1~~any valid event type),
 'Message Text' = (optional, any valid message text),
 'Notify Type' = ACK_NOTIFICATION,
 'To State' = (~~S1 or S2~~)

~~ELSE~~

~~BEFORE Notification Fail Time~~

~~RECEIVE~~

~~ConfirmedEventNotification-Request,
 'Process Identifier' = (the process identifier configured for this event),
 'Initiating Device Identifier' = IUT,
 'Event Object Identifier' = (O1 the object detecting the alarm),
 'Time Stamp' = (T2: any valid time stamp),
 'Notification Class' = (the notification class configured for this event),
 'Priority' = (the priority configured for this event type),
 'Event Type' = (E1 any valid event type),
 'Message Text' = (optional, any valid message text),
 'Notify Type' = ACK_NOTIFICATION)~~

~~14/6.~~ TRANSMIT BACnet-SimpleACK-PDU

~~15.~~ IF (Protocol_Revision is present AND Protocol_Revision >= 1) THEN

~~17.~~ RECEIVE

DESTINATION = (a device other than the TD),
 SOURCE = IUT,
 ConfirmedEventNotification-Request,
 'Process Identifier' = (~~PID~~the process identifier configured for this event),
 'Initiating Device Identifier' = IUT,
 'Event Object Identifier' = (~~O1~~the object detecting the alarm),
 'Time Stamp' = (T2),
 'Notification Class' = (~~NC~~ the notification class configured for this event),
 'Priority' = (~~P1~~the priority configured for this event type),
 'Event Type' = (~~E1~~any valid event type),
 'Message Text' = (optional, any valid message text),
 'Notify Type' = ACK_NOTIFICATION,
 'To State' = (~~S1 or S2~~)

~~ELSE~~

~~RECEIVE~~

~~DESTINATION = (a device other than the TD),
 SOURCE = IUT,
 ConfirmedEventNotification-Request,~~

~~'Process Identifier' = _____ (the process identifier configured for this event),
 'Initiating Device Identifier' = IUT,
 'Event Object Identifier' = _____ (O/the object detecting the alarm),
 'Time Stamp' = _____ (T2),
 'Notification Class' = _____ (NC the notification class configured for this event),
 'Priority' = _____ (the priority configured for this event type),
 'Event Type' = _____ (E/any valid event type),
 'Message Text' = _____ (optional, any valid message text),
 'Notify Type' = _____ ACK_NOTIFICATION~~

~~1618. TRANSMIT BACnet-SimpleACK-PDU~~

~~1719. VERIFY (the 'Event Object Identifier' from the event notification) OI, Acked_Transitions = (TRUE,TRUE,TRUE)~~

9.2 ConfirmedCOVNotification Service Execution Tests

9.2.2 Negative ConfirmedCOVNotification Execution Tests

9.2.2.1 Change of Value Notification Arrives after Subscription has Expired

Reason for Change: Added MAKE step to cancel subscription.

Purpose: To verify that an appropriate error is returned if a COV notification arrives after the subscription time period has expired.

Configuration Requirements: If the IUT does not support initiation of SubscribeCOV-Request with 'Issue Confirmed Notifications' equal to TRUE, then this test shall be skipped.

Test Steps:

1. RECEIVE SubscribeCOV-Request,
 'Subscriber Process Identifier' = _____ (any valid process identifier, P1),
 'Monitored Object Identifier' = (any object X of a type that supports COV notification),
 'Issue Confirmed Notifications' = TRUE,
 'Lifetime' = (any valid Lifetime)
2. TRANSMIT BACnet-SimpleACK-PDU
3. TRANSMIT ConfirmedCOVNotification-Request,
 'Subscriber Process Identifier' = (P1),
 'Initiating Device Identifier' = TD,
 'Monitored Object Identifier' = X,
 'Time Remaining' = (any amount of time greater than 0),
 'List of Values' = (a list of values appropriate to object X)
4. IF (the IUT can cancel the subscription) THEN
5. *MAKE (the IUT cancel the subscription)*
6. RECEIVE SubscribeCOV – Request,
 'Subscriber Process Identifier' = (PI),
 'Monitored Object Identifier' = X
- ELSE
7. MAKE (the IUT stop resubscribing, if it resubscribes automatically)
8. WAIT (at least Lifetime, but sufficient to ensure the subscription has expired)
9. TRANSMIT ConfirmedCOVNotification-Request,
 'Subscriber Process Identifier' = (P1),
 'Initiating Device Identifier' = TD,
 'Monitored Object Identifier' = X,
 'Time Remaining' = (any amount of time greater than 0),
 'List of Values' = (a list of values appropriate to object X)
10. IF (Protocol_Revision is present and Protocol_Revision >= 10) THEN
11. RECEIVE BACnet-Error-PDU,
 'Error Class' = SERVICES,

```

        'Error Code' = UNKNOWN_SUBSCRIPTION |
        (BACnet-SimpleACK-PDU)
    ELSE
12.    RECEIVE BACnet-Error-PDU,
        'Error Class' = SERVICES,
        'Error Code' = (any valid error code for class SERVICES) |
        (BACnet-SimpleACK-PDU)

```

9.16 CreatObject Service Execution Tests

9.16.2 Negative CreateObject Service Execution Tests

9.16.2.3 Attempting to Create an Object with an Object Identifier That is Not Creatable by Specifying the Object Identifier

Reason for Change: Clarify the object to be used.

Purpose: To verify the correct execution of the CreateObject service request when the 'Object Specifier' parameter conveys an object identifier for an object type that is not dynamically creatable in the IUT.

Test Steps:

1. TRANSMIT CreateObject-Request,
 'Object Identifier' = (any ~~object identifier having a~~ supported object type for which dynamic creation *using the CreateObject service* is not supported)
2. RECEIVE CreateObject-Error,
 Error Class = OBJECT,
 Error Code = DYNAMIC_CREATION_NOT_SUPPORTED
 'First Failed Element Number' = 0
3. VERIFY (the IUT's Device object),
 Object_List = (any object list that does not contain the object specified in step 1)

~~Notes to tester: If the IUT limits the instances that can be created, this shall be taken into account when selecting an object identifier in step 1.~~

9.16.2.8 Attempting to Create a non-Supported Object Type (by Object Identifier)

Reason for Change: Clarified the type of object to be used.

Purpose: To verify the correct execution of the CreateObject service request when the 'Object Specifier' parameter conveys an object identifier for an object type that is not supported in the IUT.

~~Notes to Tester: If the IUT limits the instances that can be created, this shall be taken into account when selecting an object identifier in step 1.~~

Test Steps:

1. TRANSMIT CreateObject-Request,
 'Object Specifier' = (any object identifier having an unsupported object type)
2. IF (Protocol_Revision >= 10) THEN
 RECEIVE CreateObject-Error,
 'Error Class' = OBJECT,
 'Error Code' = UNSUPPORTED_OBJECT_TYPE
 'First Failed Element Number' = 0
 ELSE

RECEIVE CreateObject-Error,
 'Error Class' = (any valid error class),
 'Error Code' = (any valid error code)
 'First Failed Element Number' = 0

3. VERIFY (the IUT's Device object, Object_List = (any object list that does not contain the object specified in step 1))

9.18 ReadProperty Service Execution Tests

9.18.1 Positive ReadProperty Service Execution Tests

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 BACnet-Confirmed-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

9.20 ReadPropertyMultiple Service Execution Tests

9.20.1 Positive ReadPropertyMultiple Service Execution Tests

9.20.1.16 ReadPropertyMultiple Array Properties

Reason for Change: Correct test issue that occurred when array length was zero.

Purpose: To verify that the IUT can execute ReadPropertyMultiple service requests when the requested property is an array, ~~when its size as well as when a single element of the array is requested. Another request is made to read an element of an array where the array index is out of range.~~

Test Concept: The TD reads the size of the array property, and then reads the first and last entries in the array. Finally, the TD reads past the end of the array and ensures that the IUT returns the correct error.

Configuration Requirement: O1 is any object in the IUT database having array property P1.

Test Steps:

1. ~~VERIFY P1=X, READ X = (O1), P1, ARRAY INDEX = 0~~
2. IF (X>0) THEN
3. TRANSMIT ReadPropertyMultiple-Request,
 'Object Identifier' = O1,
 'Property Identifier' = P1,
 'Property Array Index' = 1
4. RECEIVE ReadPropertyMultiple-ACK,
 'Object Identifier' = O1,
 'Property Identifier' = P1,
 'Property Array Index' = 1,
 'Property Value' = (V, any valid value of the correct data type for property P1)
5. TRANSMIT ReadPropertyMultiple-Request,
 'Object Identifier' = O1,
 'Property Identifier' = P1,
 'Property Array Index' = X,
6. RECEIVE ReadPropertyMultiple-ACK,
 'Object Identifier' = O1,
 'Property Identifier' = P1,
 'Property Array Index' = X,
 'Property Value' = (V, any valid value of the correct data type for property P1)
- ~~7. CHECK (V is any value of the correct data type for property P1)~~
87. TRANSMIT ReadPropertyMultiple-Request,
 'Object Identifier' = O1,
 'Property Identifier' = P1,
 'Property Array Index' = (X+1)
98. RECEIVE ReadPropertyMultiple-Error,
 'Error Class' = PROPERTY,
 'Error Code' = INVALID_ARRAY_INDEX
 | ReadPropertyMultiple-ACK,
 'Object Identifier' = O1,
 'Property Identifier' = P1,
 'Property Array Index' = X+1,
 'Property Access Error' = (
 'Error Class' = PROPERTY,
 'Error Code' = INVALID_ARRAY_INDEX)

9.22 WriteProperty Service Execution Tests

9.22.1 Positive WriteProperty Service Execution Tests

9.22.1.6 Writing NULL to Non-commandable Properties

Reason for Change: The standard was changed in PR21 to require that devices not return errors when a RelinquishNULL is written to writable non-commandable Present_Value properties.

Purpose: This test case verifies that the IUT returns a Result(+) when an attempt is made to relinquish a *writable* non-commandable *Present_Value* property.

Test Concept: Write NULL, *at a priority*, to a writable non-commandable *Present_Value* property, P1 in object O1, and verify the IUT returns a Result(+) and does not modify the property.

Test Configuration: ~~None. P1 shall be a property for which NULL is not an accepted value.~~

Test Steps:

1. READ X = (O1), P1
2. TRANSMIT WriteProperty-Request,
 'Object Identifier' = O1,
 'Property Identifier' = P1,
 'Property Value' = NULL
 'Priority' = (any valid value)
3. RECEIVE BACnet-SimpleACK-PDU
4. VERIFY (O1), P1 = X

9.22.2 Negative WriteProperty Service Execution Tests

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
4. RECEIVE BACnet-Error PDU,
 Error Class = PROPERTY,
 Error Code = PROPERTY_IS_NOT_AN_ARRAY
- ELSE
5. RECEIVE BACnet-Error PDU,
 Error Class = SERVICES,
 Error Code = INCONSISTENT_PARAMETERS
64. VERIFY (Object1), P1 = X
7. 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' = 0
8. IF (Protocol_Revision is present AND Protocol_Revision >= 4) THEN
9. RECEIVE BACnet-Error PDU,
 Error Class = PROPERTY,

Error Code = PROPERTY_IS_NOT_AN_ARRAY

ELSE

10. *RECEIVE BACnet-Error PDU,*
Error Class = SERVICES,
Error Code = INCONSISTENT_PARAMETERS
11. *VERIFY (Object1), P1 = X*

9.23 WritePropertyMultiple Service Execution Tests

9.23.2 Negative WritePropertyMultiple Service Execution Tests

9.23.2.14 Writing First Element of 'List of Write Access Specifications' with Object Access Error

Reason For Change: Replace VERIFY with READ. Simplified step 3.

Purpose: To verify the ability to correctly execute a WritePropertyMultiple service request for which the first element of the 'List of Write Access Specifications' contains a specification for an unsupported object and all writes after the first failed write attempt do not take place.

Test Concept: An attempt is made to write to a single property in two different objects. The first object is not supported. The second object is supported, and the property is writable. The objective is to verify that an appropriate error response is returned and that all writes after the first failed write attempt do not take place.

Configuration Requirements: If the IUT supports any writable scalar properties that are not commandable it shall be configured with one for use in this test. If no such properties are supported the IUT shall be configured with a writable array or commandable property and the test steps modified to account for this variation. In the test description O2 and P2 will be used to designate the writable object and property having value X used for this test. The designation ~~Bad Object~~ *BadObject* will be used to indicate an object that is not supported or not present in IUT database P1 is any valid Property Identifier.

Test Steps:

1. ~~VERIFY (O2), P2 = X~~ *READ X = (O2), P2*
2. TRANSMIT WritePropertyMultiple-Request,
'Object Identifier' = BadObject,
'Property Identifier' = P1,
'Property Value' = (any valid value of the appropriate datatype for P1 ~~this property subject to the restrictions specified in the EPICS as defined in 4.4.2~~)

'Object Identifier' = O2,
'Property Identifier' = P2,
'Property Value' = (any valid value not equal to X),
3. RECEIVE WritePropertyMultiple-Error,
'Error Class' = OBJECT,
'Error Code' = (UNKNOWN_OBJECT | *UNSUPPORTED_OBJECT_TYPE*),
'Object Identifier' = BadObject,
'Property Identifier' = P1

'Property Identifier' = P1 |

(RECEIVE WritePropertyMultiple-Error,
'Error Class' = OBJECT,
'Error Code' = *UNSUPPORTED_OBJECT_TYPE*,
'Object Identifier' = BadObject,
'Property Identifier' = P1)
4. VERIFY (O2), P2 = X

9.23.2.15 Writing First Element of 'List of Write Access Specifications' with a Write Access Error

Reason For Change: Replace VERIFY with READ.

Purpose: To verify the ability to correctly execute a WritePropertyMultiple service request for which the first element of the 'List of Write Access Specifications' contains a specification for a read only property and all writes after the first failed write attempt do not take place.

Test Concept: An attempt is made to write to two properties in a single object. The first property is supported but read only. The second property is supported and writable. The objective is to verify that an appropriate error response is returned and that all writes after the first failed write attempt do not take place.

Configuration Requirements: If the IUT supports any writable scalar properties that are not commandable, it shall be configured with one for use in this test. If no such properties are supported, the IUT shall be configured with a writable array or commandable property and the test steps modified to account for this variation. In the test description, O1 will be used to designate the object, P1 the read only property having value X, P2 the writable property having value Y used for this test.

Test Steps:

1. ~~VERIFY (O1), P1=X~~ READ X = (O1), P1
2. ~~VERIFY (O1), P2=Y~~ READ Y = (O1), P2
3. TRANSMIT WritePropertyMultiple-Request,
 - 'Object Identifier' = O1,
 - 'Property Identifier' = ~~P1~~ (P1, a read-only property in O1),
 - 'Property Value' = X,
 - 'Property Identifier' = P2,
 - 'Property Value' = (any valid value not equal to Y)
4. RECEIVE WritePropertyMultiple-Error,
 - 'Error Class' = PROPERTY,
 - 'Error Code' = WRITE_ACCESS_DENIED,
 - 'Object Identifier' = O1,
 - 'Property Identifier' = P1
5. ~~VERIFY (O1), P2=Y~~ VERIFY (O1), P1 = X
6. VERIFY (O1), P2 = Y

9.23.2.16 WritePropertyMultiple Reject Test for First Element of 'List of Write Access Specifications'

Reason for Change: Add Read steps 1&2 and comment in step 3..

Purpose: This test case verifies that if IUT does sends a Reject-PDU or Error-PDU then the write attempt for the remaining element of 'List of Write Access Specifications' do not take place.

Test Concept: Object, O1, contains a writable property, P1 and object O2, contains a writable property, P2. P1 having value X and P2 having value Y are written to the IUT but the portion of the WritePropertyMultiple specifying P1 is made invalid by omitting the 'Property Value' parameter. The value of the properties are checked to ensure they have not changed.

Test Steps:

1. READ X = (O1), P1
2. READ Y = (O2), P2
3. TRANSMIT WritePropertyMultiple-Request,
 - 'Object Identifier' = O1,
 - 'Property Identifier' = P1,
 - 'Property Value' = (this field is missing including the opening and closing tags)
 - 'Object Identifier' = O2,

- 'Property Identifier' = P2
 'Property Value' = (Any valid value not equal to Y))
4. RECEIVE WritePropertyMultiple-Error,
 'Error Class' = SERVICES,
 'Error Code' = INVALID_TAG
 'Object Identifier' = O1
 'Property Identifier' = P1) |
 (BACnet-Reject-PDU,
 'Reject Reason' = INVALID_TAG |
 MISSING_REQUIRED_PARAMETER |
 INCONSISTENT_PARAMETERS |
 INVALID_PARAMETER_DATA_TYPE)
 4. VERIFY (O1), P1 = X
 5. VERIFY (O2), P2 = Y

9.23.2.17 Writing First Element of 'List of Write Access Specifications' with a Property Access Error

Reason For Change: Replace VERIFY with READ.

Purpose: To verify the ability to correctly execute a WritePropertyMultiple service request for which the first element of the 'List of Write Access Specifications' contains a specification for an unsupported property and all writes after the first failed write attempt do not take place.

Test Concept: An attempt is made to write to two properties in a single object. The first property is not supported for this object. The second property is supported for this object and writable. The objective is to verify that an appropriate error response is returned and that all writes after the first failed write attempt do not take place.

Configuration Requirements: If the IUT supports any writable scalar properties that are not commandable, it shall be configured with one for use in this test. If no such properties are supported, the IUT shall be configured with a writable array or commandable property and the test steps modified to account for this variation. In the test description, O1 will be used to designate the object, P1 the unsupported property, and P2 the writable property having value X used.

Test Steps:

1. ~~VERIFY (O1), P2 = X~~ READ X = (O1), P2
2. TRANSMIT WritePropertyMultiple-Request,
 'Object Identifier' = O1,
 'Property Identifier' = ~~P1~~(P1, an unsupported property for object O1),
 'Property Value' = (any valid value of the appropriate datatype for P1) ~~this property subject to the restrictions specified in the EPICS as defined in 4.4.2)~~
 'Property Identifier' = P2,
 'Property Value' = (any valid value not equal to X),
3. RECEIVE WritePropertyMultiple-Error,
 'Error Class' = PROPERTY,
 'Error Code' = UNKNOWN_PROPERTY,
 'Object Identifier' = O1,
 'Property Identifier' = P1
4. VERIFY (O1), P2 = X

9.23.2.21 DateTime Non-Pattern Properties Test using WritePropertyMultiple Service

Reason for Change: Update Test Concept to include meaning of O1. Remove day of week test based on IR.

Purpose: To verify that the property being tested does not accept special date field values.

Test Concept: *O1 is the object being tested.* The property being tested, P1, is written with each of the special datetime field values to ensure that the property does not accept them. A datetime DT1 is selected which is within the range that the IUT will accept for the property. The value, V1, written to the property is the datetime DT1 with one of its fields replaced with one of the date or time special values. If the property is a complex datatype, the other fields in the value shall be set within the range accepted by the IUT. *It is a local matter whether the device accepts or rejects an invalid day of week field so it is not tested.* ~~This test shall only be applied to devices claiming Protocol Revision 11 or higher.~~

~~Notes to Tester: if P1 is an array, then a non-zero array index may be provided in the TRANSMIT and the same array index observed in the WritePropertyMultiple-Error.~~

Test Steps:

1. REPEAT SV = (year unspecified, month unspecified, day of month unspecified, ~~day of week unspecified,~~ odd months, even months, last day of month, even days, odd days, hour unspecified, minute unspecified, second unspecified, hundredths unspecified) DO {
2. TRANSMIT WritePropertyMultiple-Request,
 'Object Identifier' = O1,
 'Property Identifier' = P1,
 'Property Value' = (DT1 updated with the special value SV)
3. RECEIVE WritePropertyMultiple-Error,
 'Error Class' = PROPERTY,
 'Error Code' = VALUE_OUT_OF_RANGE,
 'Object Identifier' = Object1,
 'Property Identifier' = P1)
 | (BACnet-Reject-PDU
 'Reject Reason' = INVALID_PARAMETER_DATATYPE)
 | (BACnet-Reject-PDU
 'Reject Reason' = INVALID_TAG)
 }

9.33 Who-Is Service Execution Tests

9.33.2 Execution of Who-Is Service Requests Originating from a Remote Network

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**
3. 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

3. 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

3. 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)

9.39 General Testing Service Execution

9.39.2 Unsupported Unconfirmed Services Test

Reason for Change: Changed test to verify System_Status vs check for no reset.

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.~~

10. NETWORK LAYER PROTOCOL TESTS

10.2 Router Functionality Tests

10.2.2 Processing Network Layer Messages

10.2.2.4.3 Receiving Messages for a Busy Router

Reason for Change: Corrects problems with test in 135.1-2025 in step 3.

Purpose: To verify that the IUT rejects a message destined for a busy router.

Test Steps:

1. TRANSMIT PORT B,
 DESTINATION = LOCAL BROADCAST,
 SOURCE = R2-3,
 Router-Busy-To-Network,
 Network Numbers = 3
2. RECEIVE PORT A,
 DESTINATION = LOCAL BROADCAST,
 SOURCE = IUT,
 Router-Busy-To-Network,
 Network Numbers = 3
3. TRANSMIT PORT A,
~~10.2.2.4.3.1.1.1.1.1.~~ DA = IUT,
 SOURCE = D1A,
 DNET = 3,
 DADR = D3D,
 Hop Count = 255,
 ReadProperty-Request,
 'Object Identifier' = (any BACnet standard object),
 'Property Identifier' = (any required property of the specified object)
4. RECEIVE PORT A,
 DESTINATION = D1A,
 SOURCE = IUT,
 Reject-Message-To-Network,
 Reject Reason = 2 (router busy),
 DNET = 3

10.7 Route Binding Tests

10.7.3 Router Binding via Who-Is-Router-To-Network

Reason for Change: Modify test to allow for the case where IUT is a router that does not support initiating confirmed requests.

Purpose: To verify that the IUT can ~~initiate~~*send* requests to a remote network after the IUT uses the Who-Is-Router-To-Network Network Layer service to discover the MAC address of the router to that remote network.

Test Concept: The IUT broadcasts a Who-Is-Router-To-Network request to discover the router to the desired network. The ~~TD~~*IUT* transmits a request to a device on the remote network without performing any further form of dynamic router binding. ~~If the IUT does not support Who-Is-Router-To-Network router binding, then this test shall be omitted.~~ If the IUT cannot initiate a ReadProperty request, then another confirmed service can be substituted. *If the IUT is a router that does not support*

initiating confirmed service requests then it may forward a confirmed service request from device D4A instead. The IUT may use either the general query or specific network number query form of the Who-Is-Router-To-Network service.

Note that Clause 6.5.3 specifically mentions router binding via Who-Is-Router-To-Network and does not mention router binding by lurking and noting unsolicited I-Am-Router-To-Network messages.

Test Steps:

1. MAKE (IUT transmit Who-Is-Router-To-Network to discover the router to DNET2)
2. RECEIVE
 - DA = BROADCAST,
 - SA = IUT,
 - Who-Is-Router-To-Network,
 - | (DA = BROADCAST,
 - SA = IUT,
 - Who-Is-Router-To-Network,
 - DNET = DNET2)
3. TRANSMIT
 - DESTINATION = BROADCAST,
 - SOURCE = TD,
 - I-Am-Router-To-Network,
 - Network Numbers = DNET2
4. *IF (the IUT can initiate a ReadProperty request or any other confirmed request) THEN*
45. MAKE (IUT transmit a ReadProperty request to the D2A device on the remote network)
56. RECEIVE
 - DA = TD,
 - SA = IUT,
 - DNET = DNET2,
 - DADR = D2A,
 - Hop Count = 255,
 - BACnet-Confirmed-Request-PDU,
 - 'Service Choice' = ReadProperty-Request,
 - 'Object Identifier' = (O1, any BACnet standard object in D2A),
 - 'Property Identifier' = (P1, any required property of the specified object)
67. TRANSMIT
 - DA = IUT,
 - SA = TD,
 - SNET = DNET2,
 - SADR = D2A,
 - BACnet-ComplexACK-PDU,
 - 'Service ACK Choice' = ReadProperty-ACK,
 - 'Object Identifier' = O1,
 - 'Property Identifier' = P1,
 - 'Property Value' = (any valid value)
- ELSE -- (IUT is a router that does not support initiating confirmed service requests)*
8. TRANSMIT
 - DA = IUT,
 - SA = D4A,
 - DNET = DNET2,
 - DADR = D2A,
 - Hop Count = 255,
 - BACnet-Confirmed-Request-PDU,
 - 'Service Choice' = ReadProperty-Request,
 - 'Object Identifier' = (O1, any BACnet standard object in D2A),
 - 'Property Identifier' = (P1, any required property of the specified object)
9. RECEIVE

DA = *TD*,
SA = *IUT*,
DNET = *DNET2*,
DADR = *D2A*,
SNET = *DNET4*
SADR = *D4A*
Hop Count = (any integer *x*: $0 < x < 255$),
BACnet-Confirmed-Request-PDU,
 'Service Choice' = *ReadProperty-Request*,
 'Object Identifier' = *O1*
 'Property Identifier' = *P1*

10.

TRANSMIT

DA = *IUT*,
SA = *TD*,
DNET = *DNET4*,
DADR = *D4A*,
SNET = *DNET2*,
SADR = *D2A*
Hop Count = 254,
BACnet-Confirmed-Request-PDU,
 'Service Choice' = *ReadProperty-ACK*,
 'Object Identifier' = *O1*
 'Property Identifier' = *P1*
 'Property Value' = (any valid value)

12. DATA LINK LAYER PROTOCOLS TESTS

12.3 BACnet/IP Functionality Tests

12.3.2 BBMD B/IP Device with a Server Application

12.3.2.2 Execute Original-Broadcast-NPDU

12.3.2.2.2 Execute Original-Broadcast-NPDU (Two-hop Distribution)

Reason for Change: Possibly a typographical error because the device 'ODIUT' does not exist.

Configuration Requirements: The IUT shall be configured with a BDT that contains:

B/IP Address Broadcast Distribution Mask
 IUT 255.255.255.255
 BBMD1 255.255.255.255

When the IUT is configured for NAT, the Originating-Device in Forwarded-NPDUs that originate at the IUT, OD, is equal to the Global IP Address and Port of the IUT's Internet Router. When the IUT is not configured for NAT operation, OD is equal to the IUT.

Notes to Tester: The order of the messages transmitted by the IUT is not significant.

Test Steps:

1. TRANSMIT

DA = Local IP Broadcast,
 SA = D1,
 Original-Broadcast-NPDU,
 NPDU = Who-Is

2. RECEIVE

DA = BBMD1,
 Forwarded-NPDU,
 Originating-Device = D1,
 NPDU = Who-Is

3. IF (the IUT responds with Unicast I-Am) THEN

RECEIVE DESTINATION = D1,
 Original-Unicast-NPDU,
 NPDU = I-Am

ELSE

RECEIVE

DA = Local IP Broadcast,
 Original-Broadcast-NPDU,
 NPDU = I-Am

RECEIVE

DA=BBMD1,
 Forwarded-NPDU,
 Originating-Device = ~~ODIUT~~ IUT,
 NPDU = I-Am

12.3.6 Foreign Device Management

12.3.6.3 Foreign Device Table Timer Operations

12.3.6.3.1 Non-Zero-Duration Foreign Device Table Timer Operations

Reason for change: refining the value for Remaining-Time.

Purpose: To verify that the IUT will handle FDT timer operations: finite time Foreign Device registration, re-registration, adding grace period to the supplied Time-To-Live parameter and FDT entry clearing upon timer expiration.

Configuration Requirements: The TD shall take the role of foreign device FD2. The IUT's FDT must be empty. The Network Port object for the BACnet/IP network is NP.

Notes to Tester: The accuracy of the FDT timer shall be specified by the vendor.

Test Steps:

1. TRANSMIT
 - DA = IUT,
 - SA = FD2,
 - Register-Foreign-Device,
 - 'Time-To-Live' = 60
2. RECEIVE
 - DA = FD2,
 - SA = IUT,
 - BVLC-Result,
 - 'Result Code' = 0
3. WAIT (10 seconds)
4. TRANSMIT
 - DA = IUT,
 - SA = FD2,
 - Read-Foreign-Device-Table
5. RECEIVE
 - DA = FD2,
 - SA = IUT,
 - Read-Foreign-Device-Table-Ack,
 - B/IP address of FD2,
 - Time-To-Live = 60,
 - ~~Remaining Time = 80 minus test execution time. (50 is also acceptable if Protocol_Revision < 7)~~
 - Remaining-Time = 90 (Time-To-Live + grace period) - time since FD registration*
 - (50 is also acceptable if Protocol_Revision < 7)*
6. IF Protocol_Revision >= 17 THEN
 - VERIFY NP, BBMD_Foreign_Device_Table = ((B/IP address of FD2, 60, ~~80 90 - execution time~~ *time since FD registration*))
7. TRANSMIT
 - DA = IUT,
 - SA = FD2,
 - Register-Foreign-Device,
 - 'Time-To-Live' = 40
8. RECEIVE
 - DA = FD2,
 - SA = IUT,
 - BVLC-Result,
 - 'Result Code' = 0

9. WAIT (30 seconds)
10. TRANSMIT
 - DA = IUT,
 - SA = FD2,
 - Read-Foreign-Device-Table
11. RECEIVE
 - DA = FD2,
 - SA = IUT,
 - Read-Foreign-Device-Table-Ack,
 - B/IP address of FD2,
 - Time-To-Live = 40,
 - Remaining-Time = ~~40 minus test execution time~~ 70 (*Time-To-Live + grace period*) - *time since FD registration*
 - (10 is also acceptable if Protocol_Revision < 7)
12. IF Protocol_Revision >= 17 THEN
 - VERIFY NP, BBMD_Foreign_Device_Table = ((B/IP address of FD2, 40, ~~40 70 - execution time~~ *time since FD registration*))
13. WAIT (50 seconds)
14. TRANSMIT
 - DA = IUT,
 - SA = FD2,
 - Read-Foreign-Device-Table
15. RECEIVE
 - DA = FD2,
 - SA = IUT,
 - Read-Foreign-Device-Table-Ack,
 - (No FDT entries)
16. IF Protocol_Revision >= 17 THEN
 - VERIFY NP, BBMD_Foreign_Device_Table = ()

12.4 BACnet/IPv6 Functionality Tests

12.4.4 BBMD Tests

This group of tests verifies that a B/IPv6 device that is configured as a BACnet Broadcast Management Device (BBMD) will correctly process incoming B/IPv6 messages that pertain to BBMDs. Only devices that are configured to support BBMD functionality shall execute these tests.

Configuration Requirements: The IUT's Network Port object that represents the B/IPv6 port under test shall be configured as follows:

- BACnet_IPv6_Mode is BBMD
- BACnet_IPv6_Multicast_Address is FF02::BAC0 (Link Local Multicast Address)
- BBMD_Broadcast_Distribution_Table shall contain:

B/IPv6-address
BBMD1
BBMD2
BBMD3

Unless otherwise specified, the TD shall operate as BBMD1.

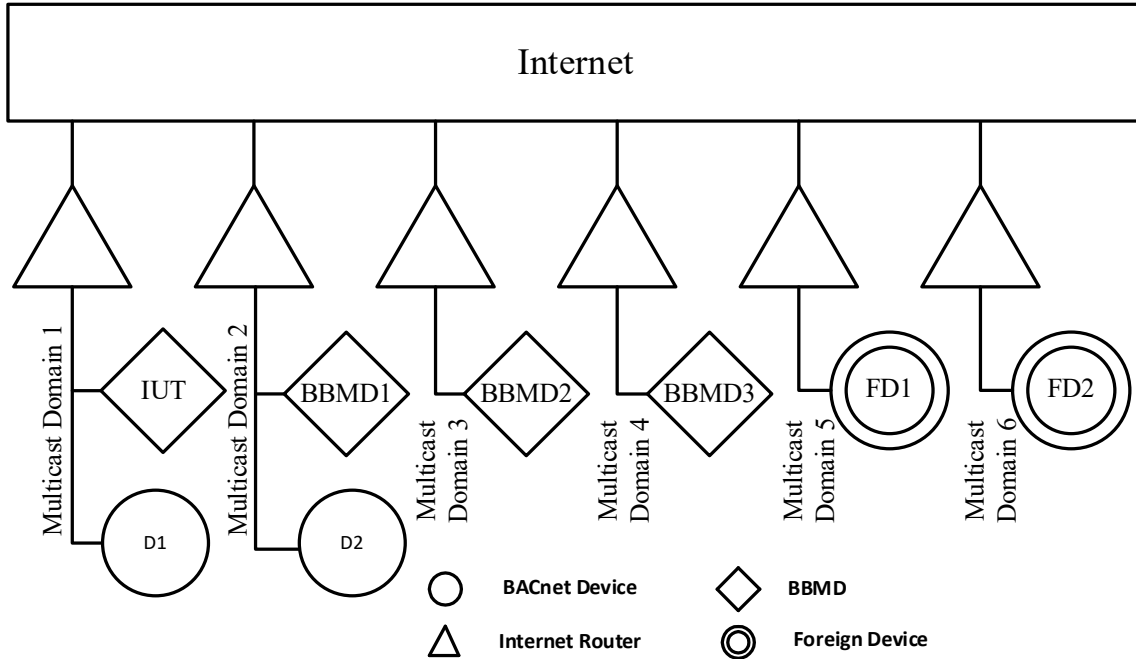


Figure 12-x Logical network configuration for BBMD tests

12.4.4.1 Positive Tests

[Delete text for this section.]

~~This group of tests verifies that a B/IPv6 device that is configured as a BACnet Broadcast Management Device (BBMD) will correctly process incoming B/IPv6 messages that pertain to BBMDs. Only devices that are configured to support BBMD functionality shall execute these tests.~~

~~Configuration Requirements: The IUT's Network Port object that represents the B/IPv6 port under test shall be configured as follows:~~

- ~~• BACnet_IPv6_Mode is BBMD~~
- ~~• BACnet_IPv6_Multicast_Address is FF02::BAC0 (Link Local Multicast Address)~~
- ~~• BBMD_Broadcast_Distribution_Table shall contain:~~

bbmd-address
BBMD1
BBMD2
BBMD3

~~For purposes of these tests, TD shall be operating as BBMD1.~~

12.4.4.1.1 Original-Broadcast-NPDU

Reason For Change: TD is not correctly defined.

Purpose: To verify that the IUT, configured as a BBMD, will forward an Original-Broadcast-NPDU request.

Configuration Requirements: The TD is a non-BBMD and on the same network as the IUT.

Test Steps:

1. TRANSMIT
 DA = B/IPv6 Link Local Multicast Address,
 SA = TD,
 Source-Virtual-Address = TD,
 Original-Broadcast-NPDU,
 Who-Is-Request
2. RECEIVE
 DA = BBMD1,
 SA = IUT,
 Forwarded-NPDU,
 Original-Source-Virtual-Address = TD
 Original-Source-B/IPv6-Address = TD
 Who-Is-Request
3. RECEIVE
 DA = BBMD2,
 SA = IUT,
 Forwarded-NPDU,
 Original-Source-Virtual-Address = TD
 Original-Source-B/IPv6-Address = TD
 Who-Is-Request
4. RECEIVE
 DA = BBMD3,
 SA = IUT,
 Forwarded-NPDU,
 Original-Source-Virtual-Address = TD
 Original-Source-B/IPv6-Address = TD
 Who-Is-Request

12.4.4.1.4 Forwarded-Address-Resolution

Reason for Change: Fix the destination address in the last step. Add Test Concept and update Configuration Requirements.

Purpose: To verify that the IUT, configured as a BBMD, will process a Forwarded-Address-Resolution request when the target virtual address is not the virtual address of the IUT.

Test Concept: FD1 shall initiate an Address-Resolution request for FD2. BBMD1 (TD) shall forward the request as a Forwarded-Address-Resolution request to the IUT. The IUT shall forward the request to FD2 and as a local multicast request.

Configuration Requirements: The TD shall operate as BBMD1 and *shall be* listed in the IUTs Broadcast Distribution Table. *FD1 shall be registered as a foreign device with the TD. FD2 shall be registered as a foreign device with the IUT.*

Notes to Tester: For Step 1 to occur, the TD must receive an Address-Resolution request from FD1 requesting the address of FD2. See below.

RECEIVE

DA = TD -- (BBMD1),
 SA = FD1,
 Address-Resolution,
 Source-Virtual-Address = FD1,
 Target-Virtual-Address = FD2

Notes to Tester: ~~The execution of step 7 is not significant but is shown here in order to demonstrate the completion of the BVLC.~~ The order of the messages ~~initiated~~ ~~transmitted~~ by the IUT is not significant. *After step 3, the FD2 would transmit an Address-Resolution-ACK to complete the BVLC transaction, but it is not part of the test. See below.*

TRANSMIT

*DA = FD1,
SA = FD2,
Address-Resolution-ACK,
Source-Virtual-Address = FD2,
Destination-Virtual-Address = FD1*

Test Steps:

1. TRANSMIT
 - DA = IUT,
 - SA = TD,
 - Forwarded-Address-Resolution,
 - Original-Source-Virtual-Address = FD1,
 - Target-Virtual-Address = FD2
 - Original-Source-B/IPv6-Address = FD1
2. RECEIVE
 - DA = B/IPv6 Link Local Multicast Address,
 - SA = IUT,
 - Forwarded-Address-Resolution,
 - Original-Source-Virtual-Address = FD1,
 - Target-Virtual-Address = FD2,
 - Original-Source-B/IPv6-Address = FD1
3. RECEIVE
 - DA = FD2,
 - SA = IUT,
 - Forwarded-Address-Resolution,
 - Original-Source-Virtual-Address = FD1,
 - Target-Virtual-Address = FD2,
 - Original-Source-B/IPv6-Address = FD1
- ~~4. TRANSMIT~~
 - ~~DA = TD,~~
 - ~~SA = FD2,~~
 - ~~Address-Resolution-ACK,~~
 - ~~Source-Virtual-Address = FD2,~~
 - ~~Destination-Virtual-Address = TD~~

12.4.5 Foreign Device Management Tests

12.4.5.1 Execute Register-Foreign-Device

Reason for change: Step 3 requested the wrong address of a non-existing device.

Purpose: To verify that the IUT will handle a Register-Foreign-Device request.

Test Steps:

1. TRANSMIT
 - DA = IUT,
 - SA = TD,

- Source-Virtual-Address = TD,
Register-Foreign-Device,
'Time-To-Live' = 60
- 2. RECEIVE
DA = TD,
SA = IUT,
Source-Virtual-Address = IUT,
BVLC-Result,
'Result Code' = 0
- 3. VERIFY NP, BBMD_Foreign_Device_Table = ((B/IPv6 address of ~~FD2~~ TD, 60, 90-execution time))

12.5 Secure Connect Functionality Tests

12.5.1 Basic Node Tests

12.5.1.1 Basic Node Positive Tests

12.5.1.1.16 Heartbeat-Request Initiation Test

Reason for Change: modified per Addendum 135-2020cc-1.

Purpose: To verify that the device initiates heartbeats as per its config.

Test Concept: With the IUT connected to the BACnet/SC network, wait the IUT's configured heart-beat interval plus 10 seconds and verify that the IUT sent a Heartbeat-Request, ensuring that no BVLCs are sent to the IUT during that period. If the IUT claims Protocol_Revision 24 or greater heartbeats are the Network Port object, SC_Heartbeat_Timeout property.

Configuration Requirements: Place the IUT in a mode where it will not initiate requests for a period longer than the heartbeat interval (except for the heartbeat request). If the IUT does not support DM-DCC-B and cannot be otherwise configured to behave in this manner, this test shall be skipped.

Test Steps:

1. REPEAT N = (1..Z) DO {
2. TRANSMIT Encapsulated-NPDU,
 'Message ID' = (M1: any valid value),
 'Originating Virtual Address' = (OVA: any valid value, including absent),
 -- 'Destination Virtual Address' absent
 'Destination Options' (absent or any valid value),
 'Data Options' = ({ X'41' }), -- Secure Path
 'BACnet NPDU' =
 ReadProperty-Request,
 'Object Identifier' = (the IUT's Device object),
 'Property Identifier' = Object_Name
3. RECEIVE Encapsulated-NPDU,
 'Message ID' = M1,
 -- 'Originating Virtual Address' absent
 'Destination Virtual Address' = OVA,
 'Destination Options' (absent or any valid value),
 'Data Options' = ({ X'41' or a list of valid header options including Secure Path }),
 'BACnet NPDU' =
 ReadProperty-ACK,
 'Object Identifier' = (the IUT's Device object),

```

        'Property Identifier' =      Object_Name,
        'Property Value' =          (the IUT's device object name)
    }
4.  BEFORE 1/2 of IUT's heartbeat interval + 10s
5.  RECEIVE Heartbeat-Request,
    'Message ID' =                  (M2: any valid value),
    -- 'Originating Virtual Address' absent
    -- 'Destination Virtual Address' absent
    'Destination Options' =        (absent or any valid value),
    -- 'Data Options' absent
6.  TRANSMIT Heartbeat-ACK,
    'Message ID' =                  M2,
    -- 'Originating Virtual Address' absent
    -- 'Destination Virtual Address' absent
    'Destination Options' =        (absent or any valid value),
    -- 'Data Options' absent

```

12.5.2 Hub Tests

12.5.2.1 Hub Positive Tests

12.5.2.1.9 Duplicate Connection Test

Reason for Change: The test mandates a specific order of connect and disconnect requests when duplicate connection requests occur. See CRR-0528.

Purpose: To verify that duplicate hub connection requests result in the original connection being dropped.

Test Concept: With the IUT operating as hub, connect device D3 to the IUT's hub URI. While maintaining this first connection, make D3 establish a second connection to the IUT's hub URI with the same VMAC. Verify that the IUT accepts the second connect request and closes the first connection. Then, while maintaining the second connection, make D3 establish another new connection (third) with a new VMAC for D3 ensure that the new connect-request is accepted and the second connection is closed.

Notes to Tester: For Steps 6, 7, and 8 and Steps 12, 13, and 14, the order in which the IUT transitions from the existing connection to the new connection is not significant.

Test Steps:

1. MAKE(D3 connect to the IUT's hub function)
2. TRANSMIT PORT (D3-IUT hub first WebSocket),
Connect-Request,
'Message ID' = (M1: any valid value),
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
-- 'Destination Options' absent
-- 'Data Options' absent
'VMAC Address' = (D3's VMAC),
'Device UUID' = (D3's UUID),
'Maximum BVLC Length' = (the D3's maximum BVLC accepted length),
'Maximum NPDU Length' = (the D3's maximum NPDU accepted length)
3. RECEIVE PORT (D3-IUT hub first WebSocket),
Connect-Accept,
'Message ID' = M1,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent

- 'VMAC Address' = (IUT's VMAC),
- 'Device UUID' = (IUT's UUID),
- 'Maximum BVLC Length' = (the IUT's maximum BVLC accepted length),
- 'Maximum NPDU Length' = (the IUT's maximum NPDU accepted length)
- 4. MAKE(D3 connect a second WebSocket to the IUT's hub function)
- 5. TRANSMIT PORT (D3-IUT hub second WebSocket),
 - Connect-Request,
 - 'Message ID' = (M2: any valid value),
 - 'Originating Virtual Address' absent
 - 'Destination Virtual Address' absent
 - 'Destination Options' absent
 - 'Data Options' absent
 - 'VMAC Address' = (D3's VMAC),
 - 'Device UUID' = (D3's UUID),
 - 'Maximum BVLC Length' = (the D3's maximum BVLC accepted length),
 - 'Maximum NPDU Length' = (the D3's maximum NPDU accepted length)
- 6. RECEIVE PORT (D3-IUT hub second WebSocket),
 - Connect-Accept,
 - 'Message ID' = M2,
 - 'Originating Virtual Address' absent
 - 'Destination Virtual Address' absent
 - 'Destination Options' = (absent or a list of valid options),
 - 'Data Options' absent
 - 'VMAC Address' = (IUT's VMAC),
 - 'Device UUID' = (IUT's UUID),
 - 'Maximum BVLC Length' = (the IUT's maximum BVLC accepted length),
 - 'Maximum NPDU Length' = (the IUT's maximum NPDU accepted length)
- 7. RECEIVE PORT (D3-IUT hub first WebSocket),
 - Disconnect-Request,
 - 'Message ID' = M3,
 - 'Originating Virtual Address' absent
 - 'Destination Virtual Address' absent
 - 'Destination Options' = (absent or a list of valid options),
 - 'Data Options' absent
- 8. TRANSMIT PORT (D3-IUT hub first WebSocket),
 - Disconnect-ACK,
 - 'Message ID' = M3,
 - 'Originating Virtual Address' absent
 - 'Destination Virtual Address' absent
 - 'Destination Options' absent
 - 'Data Options' absent
- 9. CHECK(that the IUT closed D3's initial WebSocket)
- 10. MAKE(D3 connect a third WebSocket to the IUT's hub function)
- 11. TRANSMIT PORT (D3-IUT hub second WebSocket),
 - Connect-Request,
 - 'Message ID' = (M4: any valid value),
 - 'Originating Virtual Address' absent
 - 'Destination Virtual Address' absent
 - 'Destination Options' absent
 - 'Data Options' absent
 - 'VMAC Address' = (a new VMAC for D3 which does not conflict with any other VMACs),
 - 'Device UUID' = (D3's UUID),
 - 'Maximum BVLC Length' = (the D3's maximum BVLC accepted length),
 - 'Maximum NPDU Length' = (the D3's maximum NPDU accepted length)
- 12. RECEIVE PORT (D3-IUT hub third WebSocket),
 - Connect-Accept,

- 'Message ID' = M4,
- 'Originating Virtual Address' absent
- 'Destination Virtual Address' absent
- 'Destination Options' = (absent or a list of valid options),
- 'Data Options' absent
- 'VMAC Address' = (IUT's VMAC),
- 'Device UUID' = (IUT's UUID),
- 'Maximum BVLC Length' = (the IUT's maximum BVLC accepted length),
- 'Maximum NPDU Length' = (the IUT's maximum NPDU accepted length)
- 13. RECEIVE PORT (D3-IUT hub second WebSocket),
 - Disconnect-Request,
 - 'Message ID' = M5,
 - 'Originating Virtual Address' absent
 - 'Destination Virtual Address' absent
 - 'Destination Options' = (absent or a list of valid options),
 - 'Data Options' absent
- 14. TRANSMIT PORT (D3-IUT hub second WebSocket),
 - Disconnect-ACK,
 - 'Message ID' = M5,
 - 'Originating Virtual Address' absent
 - 'Destination Virtual Address' absent
 - 'Destination Options' absent
 - 'Data Options' absent
- 15. CHECK(that the IUT closed D3's second WebSocket)

12.5.2.1.11 SC_Hub_Function_Enable Property Test

Reason for change: Fix steps 12 and 13 to correctly come from TD to IUT.

Purpose: To ensure the IUTs hub function can be enabled and disabled using the SC_Hub_Function_Enable property.

Test Concept: With the IUTs SC_Hub_Function_Enable property set to TRUE, verify the IUT is operating as a hub. Change the IUTs SC_Hub_Function_Enable property to FALSE and verify the IUT is no longer operating as a hub.

Configuration Requirements: The IUT is configured as the primary hub and the value of the SC_Hub_Function_Enable property to TRUE. The TD is configured as the failover hub. The TDs primary hub URI is configured to reference the IUT, and the IUTs failover hub URI is configured to reference the TD.

Test Steps:

1. MAKE(the TD open a WebSocket to the IUT's hub function)
2. TRANSMIT PORT (TD-IUT hub WebSocket)
 - Connect-Request,
 - 'Message ID' = (M1: any valid value),
 - 'Originating Virtual Address' absent
 - 'Destination Virtual Address' absent
 - 'Destination Options' absent
 - 'Data Options' absent
 - 'VMAC Address' = (TD's VMAC),
 - 'Device UUID' = (TD's UUID),
 - 'Maximum BVLC Length' = (the TD's maximum BVLC accepted length),
 - 'Maximum NPDU Length' = (the TD's maximum NPDU accepted length)
3. RECEIVE PORT (TD-IUT hub WebSocket)
 - Connect-Accept,

```

'Message ID' = M1,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'VMAC Address' = (IUT's VMAC),
'Device UUID' = (IUT's UUID),
'Maximum BVLC Length' = (the IUT's maximum BVLC accepted length),
'Maximum NPDU Length' = (the IUT's maximum NPDU accepted length)
4. IF (SC_Hub_Function_Enable is writable) THEN
5.   WRITE SC_Hub_Function_Enable = FALSE
6.   TRANSMIT ReinitializeDevice-Request
      'Reinitialized State of Device' = WARMSTART | ACTIVATE_CHANGES
      'Password' = (any valid password)
7.   RECEIVE BACnet-SimpleACK-PDU
    ELSE
8.   MAKE (SC_Hub_Function_Enable = FALSE)
9.   WAIT Activate Changes Fail Time
10.  CHECK(that the TD attempts and fails to open a WebSocket to the IUT)
11.  CHECK(that the IUT opens a WebSocket with the TD)
12. RECEIVETRANSMIT PORT (TD-IUTIUT-TD failover hub WebSocket)
      Connect-Request,
      'Message ID' = (M1: any valid value),
      -- 'Originating Virtual Address' absent
      -- 'Destination Virtual Address' absent
      -- 'Destination Options' absent
      -- 'Data Options' absent
      'VMAC Address' = (TD's VMAC)(IUT's VMAC),
      'Device UUID' = (TD's UUID)(IUT's UUID),
      'Maximum BVLC Length' = (the TD'sIUT's maximum BVLC accepted length),
      'Maximum NPDU Length' = (the TD'sIUT's maximum NPDU accepted length)
13. RECEIVETRANSMIT PORT (TD-IUTIUT-TD failover hub WebSocket)
      Connect-Accept,
      'Message ID' = M1M2,
      -- 'Originating Virtual Address' absent
      -- 'Destination Virtual Address' absent
      'VMAC Address' = (IUT's TD's VMAC),
      'Device UUID' = (IUT's TD's UUID),
      'Maximum BVLC Length' = (the IUT's TD's maximum BVLC accepted length),
      'Maximum NPDU Length' = (the IUT's TD's maximum NPDU accepted length)
14. VERIFY (SC_Hub_Function_Enable = FALSE)
15. IF (SC_Hub_Function_Enable is writable) THEN
16.   WRITE SC_Hub_Function_Enable = TRUE
17.   TRANSMIT ReinitializeDevice-Request
      'Reinitialized State of Device' = WARMSTART | ACTIVATE_CHANGES
      'Password' = (any valid password)
18.   RECEIVE BACnet-SimpleACK-PDU
    ELSE
19.   MAKE (SC_Hub_Function_Enable = TRUE)
20.   WAIT Activate Changes Fail Time
21.   MAKE(the TD open a WebSocket to the IUT's hub function)
22.  TRANSMIT PORT (TD-IUT hub WebSocket)
      Connect-Request,
      'Message ID' = (M1: any valid value),
      -- 'Originating Virtual Address' absent
      -- 'Destination Virtual Address' absent
      -- 'Destination Options' absent
      -- 'Data Options' absent

```

'VMAC Address' = (TD's VMAC),
 'Device UUID' = (TD's UUID),
 'Maximum BVLC Length' = (the TD's maximum BVLC accepted length),
 'Maximum NPDU Length' = (the TD's maximum NPDU accepted length)

23. RECEIVE PORT (TD-IUT hub WebSocket)

Connect-Accept,
 'Message ID' = M1,
 -- 'Originating Virtual Address' absent
 -- 'Destination Virtual Address' absent
 'VMAC Address' = (IUT's VMAC),
 'Device UUID' = (IUT's UUID),
 'Maximum BVLC Length' = (the IUT's maximum BVLC accepted length),
 'Maximum NPDU Length' = (the IUT's maximum NPDU accepted length)

12.5.3 Direct Connect Tests

12.5.3.3 Initiating Direct Connect Tests

12.5.3.3.2 Initiating Direct Connect Negative Tests

12.5.3.3.2.3 Rejection of Invalid Certificate Outgoing Connection Test

Reason for Change: Remove requirement to run the test if the accepting device does not have a valid certificate.

Purpose: To verify that the IUT will drop initiated connection attempts if the peer's certificate *has expired* ~~is invalid~~.

Test Concept: With the IUT configured to initiate direct connections. Make the IUT attempt to connect to D3 via a direct connection. D3 presents an *expired* ~~invalid~~ certificate during the *direct connect* connection attempt. Verify that the WebSocket is not established.

Configuration Requirements: The IUT is configured to initiate direct connections *to D3*. D3 is configured with *an expiring certificate* ~~an invalid certificate~~. D3 shall be configured to accept the IUT's certificate.

Test Steps:

1. IF (IUT requires D3 to be connected to the hub) THEN
2. MAKE(D3 establish a connection to the hub)
3. WAIT (for D3's certificate to expire)
4. MAKE(the IUT attempt to establish a direct connection to D3)
5. CHECK(that the IUT initiated a WebSocket connection)
6. CHECK(that the WebSocket connection was failed by the IUT)

13 SPECIAL FUNCTIONALITY TESTS

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>}
```

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?)

BACnet Testing Laboratories - Specified Tests

Version	Date	Author	Change
0.07	5-Aug-2004	Carl Neilson	Updates based on Nashville meeting comments on Round 3 updates.
0.08	24-Aug-2004	Carl Neilson	<ul style="list-style-type: none"> Removed 9.24.4.X1, 9.24.4.X2. Now exist in 135.1. Modified the purpose of 14.5.3. Modified the purpose of 14.2.2 Added 10.2.4.4
0.09		Roland Laird	<ul style="list-style-type: none"> Modified all Clause 14 tests
0.10	26-Oct-2004	Roland Laird	<ul style="list-style-type: none"> Continuation of BACnet/IP modifications - changes highlighted inline
0.11	27-Oct-2004	Carl Neilson	<ul style="list-style-type: none"> Added 7.3.1.11, 9.1.1.1, 9.1.1.4, 9.1.2.1, 9.1.2.5. The specification of the expected Time Stamp in the ack notifications was changed. - changes still highlighted inline
0.12	29-Oct-2004	Carl Neilson	<ul style="list-style-type: none"> Changes to 7.3.1.11, 9.1.1.1, 9.1.1.4, 9.1.2.1, 9.1.2.5 based on group feedback. Changes to BACnet/IP based on group feedback
0.13	23-Nov-2004	Carl Neilson	<ul style="list-style-type: none"> Added 9.24.1.X2 and 9.24.1.X3 Added "Reason For Change" to all tests. Added passing result text to 9.1.2.5 (missed in version 12) A few minor typos
0.14	20-Dec-2004	Carl Neilson	<ul style="list-style-type: none"> Added tests 10.2.2.3, 10.2.2.7.2, 10.2.2.7.3, 10.2.3.2, 10.2.3.5, 10.2.4.6, 10.2.4.8, 10.2.6 from P3-Routing-14. Modified 10.2.4.4 as per P3-Routing-14. Added tests 9.20.2.1 into RPM-B Added 7.3.2.9.8, 7.3.2.9.9, 7.3.2.17.5, 7.3.2.18.6, 7.3.2.19.5, 7.3.2.22.9 into WP-B. Added 9.23.1.X7, 9.23.1.X8 into WPM-B Added 7.3.1.X2
0.15	15-Jun-2005	Carl Neilson, Roland Laird	<ul style="list-style-type: none"> Modified 7.3.2.9.8's & 7.3.2.9.9's reason for change comments as 135.1a now incorporates the complete change. Added 8.34.X1 13.X.1 fixed test step reference in steps 8 & 18 13.4.3 changed $2 < x < 254$ to $2 < x \leq 254$ Change 9.20.2.1 to 9.20.2.X1 as the test is new and there is already a 9.20.2.1 in 135.1 Added scheduling tests 7.3.2.22.X2 and 7.3.2.22.X3 from ShedProtRev4Tests-9. Test numbers were changed to correspond with the equivalent pre-revision 4 tests.
0.16	19-Jul-2005	Carl Neilson	<ul style="list-style-type: none"> Added WhoHas tests 9.32.1.X1, 9.32.1.X2
0.17	05-Oct-2005	Jim Butler Carl Neilson	<ul style="list-style-type: none"> Added Recipient List Test 7.3.2.20.3.X1, Added MS/TP restart tests 2.2.14...2.2.17 Added RP fallback tests 8.20.Y1.X1, 8.20.Y1.X2 Added AckAlarm tests 9.1.1.X1, 9.1.1.X2 Changed 2.2.7 as per CLB-001 Changed 2.2.6 as per CLB-002 Changed 2.2.5 as per CLB-003 Changed 2.2.4 as per CLB-004 Added changes to 7.3.2.23.5
0.18	24-Oct-2005	Carl Neilson	<ul style="list-style-type: none"> Added ARCNET tests & re-arranged section 2. Added 7.3.1.X3 Array Sizing Test

BACnet Testing Laboratories - Specified Tests

			<ul style="list-style-type: none"> Added 13.X2.1 APDU Retry and Timeout
0.19	27-Oct-2005	Carl Neilson	<ul style="list-style-type: none"> Removed router qualification tests. Added reason for change to 13.X2.1 & modified note to tester Fixed incorrect numbering of BACnet/IP sections Deleted old comment as end of 7.3.2.20.3.X1
0.20	17-Jan-2006	Carl Neilson	<ul style="list-style-type: none"> Added 8.8.1 & 8.8.2 that include transmission of final BACnet-ComplexACK-PDUs
4.0.0	13-Sep-2006	Carl Neilson	<ul style="list-style-type: none"> Changed revision numbering
4.0.1	04-Apr-2007	Carl Neilson	<ul style="list-style-type: none"> Round 4 changes (excl SCHED)
4.0.2	02-May-2007	Carl Neilson	<ul style="list-style-type: none"> Added 9.10.1.X2
4.0.3	11-Jun-2007	Lori Tribble	<ul style="list-style-type: none"> Updated document per CRR-0005, CRR-0008, CRR-0009, CRR-0011, CRR-0014 Updated document per CRR-0015, CRR-0017, CRR-0020 Updated document per CRR-0021, CRR0022
4.0.4	23-Jul-2007	Lori Tribble	<ul style="list-style-type: none"> Updated the Reason For Change. Highlighted new items for Round 4 in Green Highlighted items to be deleted in Yellow. Waiting on approval of round 3 documents before we delete. Highlighted items with questions in Purple. Waiting on approval of round 3 documents before changing. See BTL Specified Tests 3.1.4 change log for details.
4.0.5	10-Oct-2007	Lori Tribble	<ul style="list-style-type: none"> Removed tests previously highlighted in yellow. These tests are now in 135.1. Added changes to tests 7.3.2.22.X1.1,2,3,4 per CRR-0030. Added changes to test 7.3.2.22.X2.3.12 per CRR-0035. Added changes to tests per WSPLab suggestions.
4.0.6	25-Oct-2007	Lori Tribble	<ul style="list-style-type: none"> Removed some of the highlighting. Updated tests per mtg 10/15/2007
4.0.7	18-Dec-2007	Lori Tribble	<ul style="list-style-type: none"> Added Virtual Routing tests Added List Manipulation Tests
4.0.8	22-Feb-2008	Lori Tribble	<ul style="list-style-type: none"> Fixed BTL-7.3.1.11 per TGTC-18.
4.0.9	01-Apr-2008	Lori Tribble	<ul style="list-style-type: none"> Updated page header format.
4.0.10	16-Apr-2008	Lori Tribble	<ul style="list-style-type: none"> Applied the following: TGTC-05 - Adds UTCTimeSync to all schedule tests. TGTC-08 - Adds 9.1.2.3 and 9.1.2.6 to this document. TGTC-13 - changes already existed in this document. TGTC-14 - Modified 7.3.2.23.9, added 7.3.2.23.10, modified 7.3.2.23.X2. TGTC-16 - Modified 7.3.1.13 TGTC-17 - 7.3.2.23.10 has already been modified by TGTC-14 TGTC-18 - Added 7.3.1.10.X1, TGTC-19 - Modified 9.21.1.4 TGTC-21 - Modified 7.3.2.22.X1.2 and 7.3.2.22.X1.4 TGTC-35 - Added 7.3.1.3. TGTC-36 - Added 7.3.1.10 TGTC-37 - Modified 7.3.1.11 TGTC-40 - Added 9.22.2.4 TGTC-41 - Modified 7.3.2.23.6.1 TGTC-43 - Added 8.22.1, 8.22.2. Modified 8.22.X2.

BACnet Testing Laboratories - Specified Tests

			<p>BTL-CRR-0018 - Modified 9.10.2.1</p> <p>BTL-CRR-0050 - Modified 7.2.1.10</p> <p>BTL-CRR-0051 - Modified 13.X1.3, 13.X1.6, 13.X1.7</p> <ul style="list-style-type: none"> Updated reason for change on several tests. Updated tests for Rev 5 and 6 Added 9.1.1.X3 Added 9.1.2.3, 9.1.2.4, 9.1.2.7 Added 13.X5.1, 13.X5.2, 13.X5.3, 13.X5.5, 13.X5.6 for Backup and Restore Initiation testing.
4.0.11	April 16, 2008	Lori Tribble	<ul style="list-style-type: none"> Accepted Changes made above Updated table of contents Added Reason for Change to tests that did not have it. Marked test 9.10.2.1 for further review
4.0.13	May 21, 2008	Lori Tribble	<ul style="list-style-type: none"> Added test correction for 9.33.2.3 per BTL-CRR-0055. Added test corrections for 9.14.2.3 and 9.15.2.2 per WS-038-4. Added test corrections for 8.4.1, 8.4.2, 8.4.3.1, 8.4.3.2, 8.4.4, 8.4.5, 8.4.6 per BTL-CRR-0017. Corrected reason for change on several tests. Removed : from test numbers
4.0.14	June 20, 2008	Lori Tribble	<ul style="list-style-type: none"> Updated tests 7.3.2.22.X1 and 7.3.2.22.X2.3.1 to match recent changes made in TI-WG on SED-004 and SED-006. Updated all tests which use the UTCTimeSynchronization service to indicate using a UTC date. Updated non-router tests per CN-092-04 Question about test 8.4.6 correction to be answered. See comment.
4.0.15	September 9, 2008	Lori Tribble	<ul style="list-style-type: none"> Applied BTL-CRR-0056 Time Master changes Applied BTL-CRR-0064 NonRouterNetworkCommands Updated document to reference 135.1-2007 section numbers. Added 7.3.2.8.1 and 7.3.2.8.3. These required updates for UTCTimeSynchronization. Added 7.3.2.21.3.1 and 7.3.2.21.3.2. These required updates to include UTCTimeSynchronization. Added 7.3.2.23.1 and 7.3.2.23.2. These required updates to include UTCTimeSynchronization. Added 7.3.2.23.3.1 - 9 and 7.3.2.23.4 - 8. These tests required updates to include UTCTimeSynchronization.
4.0.16	September 17, 2008	Lori Tribble	<ul style="list-style-type: none"> Accepted all changes made previously. Made format changes.
4.0.18	Oct 21, 2008	Lori Tribble	<ul style="list-style-type: none"> Updated Reason For Change for all tests that now have SSPC proposals. Added client side schedule tests Removed test 9.23.1.X Changed COV test from 24 hour lifetime to 8 hour lifetime.
5.0.1	Oct 21, 2008	Lori Tribble	<ul style="list-style-type: none"> Accepted all changes made above. Changed version to 5.0.1 Updated Reason For Change for tests that now have SSPC proposals.

BACnet Testing Laboratories - Specified Tests

			•
5.0.2	Feb 24, 2009	Lori Tribble	<ul style="list-style-type: none"> • Added place holder for new test BTL-8.22.X4 Writing Array properties as a Whole array. • Renumbered test steps for 7.3.2.21.3.2 • TGTC-57: Updated Configuration Requirements for test 7.3.2.23.X2.3.12 Revision 4 Lower Event Priority Change Test. • TGTC-58: Updated test 7.3.2.23.X2.3.10 Revision 4 Calendar Entry WeekNDay Odd-Numbered Month Test. • TGTC-59: Updated test 7.3.2.24.1 Log_Enable Test. • TGTC-60: Updated test 8.4.2 CHANGE_OF_STATE Tests • TGTC-79: Updated tests 8.2.1 through 8.2.8 to include BEFORE Notification Fail Time before each notification. • TGTC-80: added tests 9.10.1.1 through 9.10.1.3 to wait Notification Fail Time before each notification. • TGTC-81: Added test 7.2.2. • TGTC-84: Updated test 7.3.2.24.10 Notification_Threshold Test • TGTC-85: Updated test 8.4.7 BUFFER_READY Tests • BDS-001: Updated tests 7.3.2.23.5 Exception_Schedule Restoration Test and 7.3.2.23.6 Weekly_Schedule Restoration Test • BTL-CRR-0069: Updated test 10.X.1 Static Router Binding, 10.X.2 Router Binding via Application Layer Services, 10.X.3 Router Binding via Who-Is-Router-To-Network, and 10.X.4 Router Binding via Broadcast. • BTL-CRR-JN3: Updated test 2.2.7. • Added database revision tests from 135.1-2007f.
5.0.4	27-Mar-2009	Lori Tribble	<ul style="list-style-type: none"> • Changed test 9.2.1.X8 to be 9.3.X9. The test doesn't exist yet but is supposed to be the unconfirmed version of the 9.2.1.X4 test which is also not written. • Added place holder for 8.4.X2 Extended Algorithm Tests (ConfirmedEventNotification) and 8.5.X3 Extended Algorithm Tests (UnconfirmedEventNotification). • Renumbered 9.23.2.X7 to 9.23.2.6 (as defined in 135.1-2007) • Renumbered 9.23.1.X8 to 9.23.2.7 (as defined in 135.1-2007) • Updated tests 9.14.2.3 and 9.23.2.6 per BTL-CRR-0072 • Updated tests 9.1.1.1 and 9.1.1.4 per TGTC-111 • Updated test 7.3.2.23.X2.3.9 per TGTC-127 • Updated test 7.3.2.21.3.X per TGTC-128 • Updated test 9.22.1.X2 per TGTC-133 • Added test 7.3.2.24.8 per BTL-CRR-0070 • Added Chapter 6 sections that are changed or new to 135.1 and whose contents are being used within some of the tests (i.e. READ) (Described in CN-093) • Added section 7.2.1.3 to document to show proposed change to text. (FR-??)
5.0.5	6-Apr-2009	Lori Tribble	<ul style="list-style-type: none"> • Added reason for change to chapter 6 section and for test 7.2.2.

BACnet Testing Laboratories - Specified Tests

			<ul style="list-style-type: none"> Modified text for 7.2.1.3 and added reason for change. Added reason for change for the Record_Count test (7.3.2.24.8)
5.0.6	9-Apr-2009	Lori Tribble	<ul style="list-style-type: none"> Added test 9.2.20.1 Reading a Single, Unsupported Property from a Single Object. Per CRR-0039. Fixed spelling error in Configuration Requirements of Stop_When_Full TRUE Test (7.3.2.24.6.1). Updated Create and Delete Tests per proposal provided to BTL-WG and approved on 4/9/2009. Tests modified are: 8.16.2, 8.16.3, 8.16.4, 9.16.1.1, 9.16.1.2, 9.16.1.3, 9.16.1.4, 9.16.2.1, 9.16.2.2, 9.16.2.3, 9.16.2.4, 9.16.2.5, 9.16.2.6, 9.17.1.1. Also removed test 9.16.1.X1 per this document.
5.0.7	8-Jun-2009	Lori Tribble	<ul style="list-style-type: none"> Added to 9.20.2.1 that this change is included in CN-121. Changed test 7.3.1.1 per BTL-CRR-0074 and DJH-001-3.
5.0.8	22-Jun-2009	Lori Tribble	<ul style="list-style-type: none"> Removed test 9.23.1.7 Writing Maximum Multiple Properties test. Updated test 7.3.1.11 to update the configuration requirements to include initial configuration of the ACK_Required property. Test 7.3.2.23.X1.1 - updated configuration requirements Updated tests 7.3.2.23.7, 7.3.2.23.8, 7.3.23.X2.8, 7.3.23.X2.7 step 1 to correctly reference D_i not D₁ Updated test 7.3.1.10 configuration requirements. Changed 'read-only' to 'not configurable'. Removed all highlighting Updated TOC.
5.0.final	26-Jun-2009	Lori Tribble	<ul style="list-style-type: none"> Accepted all changes per acceptance by BTL-WG 6/18/2008.
6.0.1	26-Jan-2011	Duffy O'Craven	<ul style="list-style-type: none"> Corrected: 'inside' for 'outside' in step 4 of test 7.3.2.8.2, based upon BTL-CRR-0172_7.3.2.8.2_inside_outside.doc Put section 7.3.2.10 in order, before 7.3.2.21 Revised test 7.3.2.23.X2.4 Revision 4 Weekly_Schedule and Exception_Schedule Interaction Test, based upon KV-001-03_7.3.2.23.X2.4.doc Adjusted heading on test instead of section, so that test 7.3.2.24.6.1 appears in Table of Contents. Added tests 9.1.1.X4 and 9.1.1.X5 to ACK-B, based upon BTL Specified Tests-Add135-2004m-4-ReAckAlarms-3.doc Removed test 9.1.2.6, as the correct version is now in 135.1-2009, per BTL-CRR-0125_9.1.2.6.doc Added test 9.21.1.X5 Reading Items with Negative Count and MOREITEMS Derived tests from 135.1-2009 in DCC-A and RD-A, adding proper password treatment based upon BTL-CRR-0078 DeviceCommunicationControl_Password.doc Revised tests 9.24.2.1, 9.24.2.2, 9.27.1.1 and 9.27.1.3, and added tests 9.24.2.X3, 9.27.2.X3 and 9.27.2.X4 in

BACnet Testing Laboratories - Specified Tests

			DCC-B and RD-B, based upon 135-2004m-8 r2 Clarify DeviceCommunicationControll and ReinitializeDevice interactions.doc
9.0.3	7-Apr-2011	Duffy O'Craven	<ul style="list-style-type: none"> • Incorporated BTL - 7.3-MO_V9.doc including new test 7.3.2.24.X7 • Derived BTL – 7.3.1.12 with modifications in consequence of BTL-CRR-0171_7.3.1.12_TO-NORMAL.doc • Incorporated changes to genericize tests for logging objects in BTL - 8.21-MO V8.doc and BTL - 9.21-MO V7.doc • Incorporated DO-016-08_Verify_Notification_Logging.doc as tests 7.3.2.26.X1, 7.3.2.26.X2, 7.3.2.26.X3, and 7.3.2.26.X4 • Incorporated 135-2004b-5 - Restart Parameters_v2.doc in test 8.3.X1
9.0.4	26-May-2011	Duffy O'Craven	<ul style="list-style-type: none"> • Incorporated BTL-CRR-0082_ReadOnlyTest.doc in test 7.2.2.1 • Incorporated BTL-CRR-0083_nonDocumentedProperty.doc in test 7.2.2.X2 • Added test 2.2.18 Verify Tno_token w/ Serial Analyzer in consequence of BTL-CRR-0085-NewMSTPTTest.doc • Specified Protocol_Revision ≥ 7 in test 13.2 in consequence of BTL-CRR-0087_TimeMasterTest.doc • Added modified test 9.7.1.1, in consequence of BTL-CRR-0089_9.7.1.1.doc
9.0.5	31-May-2011	Duffy O'Craven	<ul style="list-style-type: none"> • Modified tests 8.2.2, 8.2.4, 8.2.6, and 8.2.8 in consequence of BTL-CRR-0095_changeable_Status_Flags.doc • Modified tests 8.4.4, 8.4.5, 8.4.6, and 8.4.7 in consequence of BTL-CRR-0096_object_referenced_by_EE.doc • Specified Protocol_Revision < 10, in consequence of BTL-CRR-0104_correcting_9.10.2.1.doc • Corrected Test Concept: from TD to IUT in 10.X.3 in consequence of BTL-CRR-0106_10.X.3_TestingHints.doc • Corrected Test Configuration: of test 7.3.2.24.4 in consequence of BTL-CRR-0116_Log_Interval_read-only.doc • Corrected expected result in test 9.14.2.2 in consequence of BTL-CRR-0117_9.14.2.2_First_Failed_Element.doc • Corrected the name of test 9.30.1.1, and added BTL Specified Tests versions of 9.30.1.2, 9.31.1.1 and 9.31.1.2 derived from 135.1 - 2007 as specified in BTL-CRR-0113_9.31.1.1 diverge dissimilar tests.doc
9.0.6	09-Jun-2011	Duffy O'Craven	<ul style="list-style-type: none"> • Added tests 7.3.2.29.X1 and 7.3.2.29.X2 for Structured View in consequence of Structured View Test Plan v6.doc
9.0.7	12-Jun-2011	Duffy O'Craven	<ul style="list-style-type: none"> • Revised test 7.2.2.X2 to restrict the test to standard object types, in consequence of BTL-CRR-0130_7.2.2.X2.doc and making it identical to the revision in 135.1-2009n-1

BACnet Testing Laboratories - Specified Tests

			<ul style="list-style-type: none"> • Further revised test 7.2.2.X2, in consequence of BTL-CRR-0180_P_C_C.doc • Further revised test 7.2.2, in consequence of BTL-CRR-0178_allowed-values_REAL.doc • Modified test 10.X.5 to ensure that the packet actually reaches the IUT, and that the test uses an address which resembles the actual address of IUT, in consequence of BTL-CRR-0138_10.X.5_same_DADR.doc • Removed test 7.3.2.21.3.X, as the version in 135.1-2009g-6 replaces it, in consequence of BTL-CRR-0141_7.3.2.21.3.X_DDB_without_range.doc. • Removed tests 8.22.1 and 8.22.2, as 135.1-2009i-7 ratified the Notes to Tester: addition that had caused these revised tests to supercede the 135.1 - 2003 - 8.22.1 and 135.1 - 2003 - 8.22.2 versions. • Removed test 8.22.X1, as the version in 135.1-2009i-8 replaces it. • Added qualifying language in Notes to Tester: of tests 7.3.2.23.X1.3, 7.3.2.23.X1.4, and 7.3.2.23.X2.8 in consequence of BTL-CRR-0158_Sch_Object_Writes.doc • Revised test 10.X.2 in consequence of BTL-CRR-0149_non-BROADCAST.doc • Removed test 14.1.7 in consequence of BTL-CRR-0152_eliminating_14.1.7.doc • Added to the Configuration Requirements: of test 7.3.2.24.X3, in consequence of BTL-CRR-0151_7.3.2.24.X3.doc • Adds a Notes to Tester: to test 7.3.2.24.X1, in consequence of BTL-CRR-0160_Log-interrupted.doc • Further revised test 7.3.2.24.8 in consequence of BTL-CRR-0169_7.3.2.24.7_Not_all_at_once.doc • Derives with modifications, and adds a Notes to Tester: to create test BTL - 7.3.2.24.12, in consequence of BTL-CRR-0165_7.3.2.24.12.doc • Added 'Server' = TRUE, to test 7.1 and derived a BTL Specified Test 13.1.12.1 with that change, in consequence of BTL-CRR-0177_server_in_Abort-PDU.doc • Further revised test 9.1.2.3, and derived test BTL - 9.1.2.6 in consequence of BTL-CRR-0195_9.1.2.3_and_9.1.2.6.doc • Further revised tests 9.10.1.1 and 9.10.1.2, in consequence of BTL-CRR-0182_9.10.1.2.doc • Further revised test 9.10.1.1, and derived test BTL - 9.10.1.7, in consequence of BTL-CRR-0194_ACK_in_9.10.1.1_and_9.10.1.7.doc • Further derived 9.10.1.7, in consequence of BTL-CRR-0184-9.10.1.7.doc and BTL-CRR-0200-9.10.1.7.doc • Removed test 9.21.1.4 as 135.1-2009g-16 replaced it, in consequence of BTL-CRR-0201_9.21.1.4.doc
9.0.8	22-Jun-2011	Duffy O'Craven	<ul style="list-style-type: none"> • Referred from BTL 7.2.2.X2 to test 7.1.x and from BTL 7.2.2.1 to test 7.2.x in 135.1-2009i-22, which is the first

BACnet Testing Laboratories - Specified Tests

			<p>occurrence of this test in a ratified addendum., but with slightly different content.</p> <ul style="list-style-type: none"> • Added qualifying language in Notes to Tester: of tests 7.3.2.23.8 in consequence of BTL-CRR-0158_Sch_Object_Writes.doc • Changed test numbers for tests 7.3.2.23.X2.1 through 7.3.2.23.X2.8 (now 7.3.2.23.X.1 through 7.3.2.23.X.8), 7.3.2.23.X2.3.1 through 7.3.2.23.X2.3.13 (now 7.3.2.23.X.3.1 through 7.3.2.23.X.3.13), and 7.3.2.23.X1 through 7.3.2.23.X4 (now 7.3.2.23.Y.1 through 7.3.2.23.Y.4) and 7.3.2.23.X3 (now 7.3.2.23.Y) to the 135.1-2009j-17 test number used for BUFFER_READY tests. • Changed test number for 8.5.X1 to the 8.5.7 used in 135.1-2009l. • Added more qualifying language in Notes to Tester: of tests 7.3.2.23.X1.3, 7.3.2.23.X1.4, and 7.3.2.23.8 incorporating from the versions in 135.1-2009g-17, 135.1-2009g-21 and 135.1-2009i-7 • Added mention of the version in 135.1-2009j-14, in test 7.3.2.24.4 The BTL Specified Test takes precedence. • Added references to the version in 135.1-2009i-14, in tests 7.3.2.24.X1, 7.3.2.24.X2, and 7.3.2.24.X3 The BTL Specified Tests take precedence. • Added references to the version in 135.1-2009i-14, in tests 7.3.2.24.X4, 7.3.2.24.X6, and 7.3.2.24.X7 The versions of these tests are identical with those in BTL Specified Tests. • Replaced test 7.3.2.24.5 with exactly the 135.1-2009g-16 version, adjusting only some capitalization typos, with no semantic difference. • Added mention of the version in 135.1-2009j-10, in tests 7.3.2.24.6.1, and 7.3.2.24.6.2 The versions of these tests are quite different. • Replaced tests 7.3.2.24.7, and 7.3.2.24.8 with exactly the 135.1-2009j-13 and 135.1-2009j-14 PPR1_DRAFT versions, with no semantic difference. • Added references to the versions in 135.1-2009h-3, in tests 8.2.1 through 8.2.8 • Added reference to the version in 135.1-2009i-3, in test 8.2.x1 which is identical with the version in BTL Specified Tests. • Replaced tests 8.18.1, 8.18.2, 8.18.X1, and 8.18.X2 with exactly the 135.1-2009i-4 versions, which compared with the prior BTL Specified version means adjusting a syntactically incorrect VERIFY to CHECK, with no semantic difference. • Specified (BACnetDeviceObjectPropertyReference—referring to the buffer property of the log object) as the first part of Event_Values in every ConfirmedEventNotification of BUFFER_READY event type.
9.0.9	6-Jul-2011	Duffy O'Craven	<ul style="list-style-type: none"> • Removed tests 7.3.2.10.X3, 7.3.2.10.X4, 7.3.2.10.X5 as these are identical to the versions in 135.1-2009f-2.

BACnet Testing Laboratories - Specified Tests

			<ul style="list-style-type: none"> Renumbered test 7.3.2.10.X6 to 7.3.2.10.X4 to match the , number of the corresponding test which is in 135.1-2009f-2, and which is identical, except for an errata, with the version in BTL Specified Tests.
9.0.10	1-Aug-2011	Duffy O'Craven	<ul style="list-style-type: none"> Fixed a type "end" for "and", in test 13.1.X6 Fixed step number reference in step 14 of tests 9.1.1.X4 and 9.1.1.X5 per BTL-CRR-0214 9.1.1.X4 and 9.1.1.X5.doc
9.0.11	28-Sep-2011	Duffy O'Craven	<ul style="list-style-type: none"> Incorporated DO-014-01_TimeMaster.doc as tests 13.2.1 through 13.2.7 Eliminated Chapter 6 Conventions for Specifying BACnet Conformance Tests, since that content is now completely expressed in 135.1-2009 Corrected the missing underscore typo in Record_Count in test 7.3.2.24.X7, and renumbered the steps to be consecutive.
9.0.12	30-Sep-2011	Duffy O'Craven	<ul style="list-style-type: none"> Deleted tests 7.3.1.11, 9.1.1.1, 9.1.1.4, 9.1.2.1 and 9.1.2.5, as the versions from 135.1-2009f-1 take precedence. Deleted tests 10.2.2.3, 10.2.2.7.2, 10.2.3.2, 10.2.3.5, 10.2.4.4, 10.2.4.6, 10.2.4.8, and 10.2.6 as the versions from 135.1-2009g-3 take precedence. Deleted tests 9.1.1.X1 and 9.1.1.X2 as the versions in 135.1-2009g-4 take precedence. Deleted test 12.1.1.9.X1 because it is identical to test 12.1.1.9.X in 135.1-2009g-5 Deleted tests 9.24.1.X2 and 9.24.1.X3 as the versions in 135.1-2009g-8 with numbers 9.24.1.X1 and 9.24.1.X2 take precedence. Deleted test 7.3.1.1 as the version in 135.1-2009g-9 takes precedence. Deleted tests 10.X.1, 10.X.2, 10.X.3, and 10.X.4 as the versions in 135.1-2009g-10 - 10.Y.1, 10.Y.2, 10.Y.3, and 10.Y.4 take precedence. Deleted tests 10.X.5, 10.X.6, and 10.X.7 as the versions in 135.1-2009g-10 - 10.X.1, 10.X.2, and 10.X.3 take precedence. Added tests 7.3.2.21.1, 7.3.2.21.3.4, and 8.4.8.14 as the versions in 135.1-2009g-11 only portrayed the intended revision with a context-diff, so the entirety of the revised tests is rendered here. Modified test 8.4.2 with the change in 135.1-2009g-11 Deleted tests 8.18.X3, 8.22.X2, and 8.22.X3 as the versions in 135.1-2009g-14 - 8.18.3, 8.22.4, and 8.22.5 take precedence. Deleted tests 9.4.X1, 9.4.X2, 9.5.X1, and 9.5.X2 as the versions in 135.1-2009g-15 - 9.4.5, 9.4.6, 9.5.1, and 9.5.2 take precedence.
9.0.13	10-Oct-2011	Duffy O'Craven	<ul style="list-style-type: none"> Deleted test 7.3.2.24.9 as the version in 135.1-2009g-16 takes precedence. Deleted tests 7.3.2.23.3.1, 7.3.2.23.X.3.1, 7.3.2.23.X.3.2, 7.3.2.23.X.3.3, 7.3.2.23.X.3.4, 7.3.2.23.X.3.5, 7.3.2.23.X.3.6 as the versions in 135.1-2009g-17 take precedence. Note that the test numbers used in 135.1-

BACnet Testing Laboratories - Specified Tests

			<p>2009g-17 each specify X rather than the X2 used in Test Plan-5.0.final and BTL Specified Test-5.0.final.</p> <ul style="list-style-type: none"> Deleted tests 13.X3 and 13.X4 as the 13.X1 and 13.X2 versions in 135.1-2009g-19 take precedence. Deleted tests 8.3.X1 and 9.3.X8 as the versions 8.3.X and 9.3.1 in 135.1-2009g-20 take precedence. Deleted tests 7.3.2.23.Y.1, 7.3.2.23.Y.2, 7.3.2.23.Y.3, and 7.3.2.23.Y.4 as the versions in 135.1-2009g-21 take precedence. Note that the test numbers used in 135.1-2009g-21 each specify Y rather than the X1 used in Test Plan-5.0.final and BTL Specified Test-5.0.final. Corrected COLDSTART to WARMSTART in test 7.3.2.23.5 in accordance with 135.1-2009i-1 Deleted tests 8.8.1 and 8.8.2 as the versions in 135.1-2009i-5 take precedence. Deleted tests 8.20.Y1.1 and 8.20.Y1.2 as the versions in 135.1-2009i-6 take precedence.
9.0.14	14-Nov-2011	Duffy O'Craven	<ul style="list-style-type: none"> Fixed the number on test 9.16.1.2 (was inadvertently 16.1.1.2 in BTL Specified Tests-5.0.final.) Put test 13.X6.5.1 in the Table of contents, by giving it Header 4 style. Removed the Notes to tester: section of test 7.3.1.11 which had had the rest of the test removed in revision 9.0.12. Separated the Purpose and Test Concept of test 7.3.1.13. Fixed the indentation of step 14. In test 7.3.1.13 Removed test 7.3.1.X1 as it is identical to the version in 135.1- 2009d-2 - 7.3.2.10.1 Added Reason for change (to correct a cut&paste&forgot-to-revise typo in the Test Concept) to test 7.3.2.10.X4 Added Reason for change (the version in 135.1-2009g-11 only portrays the intended revision with a context-diff, so the entirety of the revised test is rendered here) to test 7.3.2.21.3.4 Removed test 7.3.1.X2 as it is identical to the version in 135.1- 2009i-15 - 7.3.2.11.X Removed test 7.3.2.21.X1 as it is identical to the version in 135.1- 2009g-7 - 7.3.2.20.X (note that is the second test in that addenda with that same number, there is another in g-6). Removed tests 9.1.1.X1 and 9.1.1.X2 as the versions in 135.1-2009g-4 take precedence.
9.0.15	23-Nov-2011	Duffy O'Craven	<ul style="list-style-type: none"> Removed test 8.34.X1 as it is identical to the version in 135.1- 2009i-12. Removed tests 9.1.1.X4 and 9.1.1.X5 as the versions in 135.1-2009i-17 take precedence. Removed test 9.10.1.X2 as the version in 135.1-2009d-1 - 9.10.X takes precedence. Added Notes to tester: to tests 9.14.2.2 and 9.14.2.3 in consequence of BTL-CRR-0232_9.14.2.2_addl_error_codes.doc, and also applied Protocol_Revision conditional from the version in 135.1-2009i-10 to test 9.14.2.3.

BACnet Testing Laboratories - Specified Tests

			<ul style="list-style-type: none"> Removed test 8.16.2 because the correction has already been applied in 135.1-2007. Removed tests 8.16.3, 8.16.4, 9.16.1.1, 9.16.1.3, 9.16.2.2, 9.16.2.3, 9.16.2.4, and 9.16.2.5 because the versions in 135.1-2009f-3 take precedence. Note that BTL - 9.16.1.4 is preserved for it contains a more accurate restriction of “...any unique object identifier of a type that is creatable <i>and an instance number that is creatable</i>”. Removed tests 9.21.1.1, 9.21.1.2, 9.21.1.3, 9.21.1.4.X1, 9.21.1.6.X1, 9.21.1.6.X2, 9.21.1.X1, 9.21.1.X2, and 9.21.2.X4 because the versions in 135.1-2009i-14 take precedence. Note that BTL - 9.21.1.X3 is preserved for it contains a more accurate list: “Qualifying tests are: 9.21.1.1, 9.21.1.2, 9.21.1.3, 9.21.1.4, 9.21.1.4.X1, 9.21.1.X1 or 9.21.1.X2.” Removed test 9.23.2.6 as the version in 135.1-2009i-10 takes precedence. Removed test 9.20.2.1 as the version in 135.1-2009i-11 takes precedence. Removed tests 13.X3 and 13.X4 as the versions in 135.1-2009g-19 take precedence. Test WARMSTART with no Password is made 9.27.1.3, in correspondence with 135.1-2007 numbering. Removed entire Chapter 14, replicated in 135-2009e-1
9.0.final	01-Dec-2011	Duffy O’Craven	<ul style="list-style-type: none"> Updated from 9.0.15 to 9.0.final, accepting all change tracking
12.0.1	25-Jul-2012	Lori Tribble	<ul style="list-style-type: none"> Applied Errata 9.0 7/19/2012 Applied Addendum 9.0-a Applied Addendum 9.0-b Applied Addendum 9.0-c Applied Errata 12.0 7/23/2012
12.0.2	02-Aug-2012	Lori Tribble	<ul style="list-style-type: none"> Applied Errata-BTL Test Package 9.0 plus addenda 8/02/2012 (includes above Errata which was not published)
12.0.final	02-Aug-2012	Lori Tribble	<ul style="list-style-type: none"> Accepted all changes and Changed Name
12.1.1	27-Sept-2013	Lori Tribble	<ul style="list-style-type: none"> Applied Addendum 12.0b
12.1.2	27-Sept-2013	Lori Tribble	<ul style="list-style-type: none"> Applied Addendum 12.0c
12.1.3	30-Sept-2013	Lori Tribble	<ul style="list-style-type: none"> Applied Addendum 12.0d
12.1.4	30-Sept-2013	Lori Tribble	<ul style="list-style-type: none"> Applied Addendum 12.0e
12.1.5	1-Oct-2013	Lori Tribble	<ul style="list-style-type: none"> Applied Addendum 12.0f
12.1.6	1-Oct-2013	Lori Tribble	<ul style="list-style-type: none"> Applied Addendum 12.0g
12.1.7	1-Oct-2013	Lori Tribble	<ul style="list-style-type: none"> Applied Errata 9/30/2013
14.0.a	1-Nov-2014	Lori Tribble	<ul style="list-style-type: none"> Applied Addendum 12.1a
14.0.b	1-Nov-2014	Lori Tribble	<ul style="list-style-type: none"> Applied Addendum 12.1b
14.0.c	1-Nov-2014	Lori Tribble	<ul style="list-style-type: none"> Applied Addendum 12.1c
14.0.d	1-Nov-2014	Lori Tribble	<ul style="list-style-type: none"> Applied Addendum 12.1d
14.0.e	1-Nov-2014	Lori Tribble	<ul style="list-style-type: none"> Applied Addendum 12.1e
14.0.plus_errata	3-Nov-2014	Lori Tribble	<ul style="list-style-type: none"> Updated Reason for Change on all remaining tests. Removed some tests which existed in 135.1-2013.
14.0.final	19-Nov-2014	Duffy O’Craven	<ul style="list-style-type: none"> Removed comments, and pdated to 14.0.final without change.
15.0.05	24-Aug-2017	Lori Tribble	<ul style="list-style-type: none"> Applied Addenda 14.0b-j plus errata

BACnet Testing Laboratories - Specified Tests

15.0.08	25-Sep-2017	Lori Tribble	• Removed test 8.4.X9.
15.0.11	11-Oct-2017	Lori Tribble	• Applied errata from voting members
15.0.final	11-Oct-2017	Lori Tribble	• Accepted all changes
15.1.1	30-Mar-2018	Lori Tribble	• Applied addenda a, b , c, d, and errata
15.1.2	6-Apr-2018	Lori Tribble	• Accepted all changes
15.1.3	26-Apr-2018	Lori Tribble	• Reformatted almost all tests to meet 135.1 formats, applied errata
15.1.4	1-May-2018	Lori Tribble	• Applied Errata
15.1.5	3-May-2018	Lori Tribble	• Fixed formatting and numbering issues.
15.1.final	1-June-2018	Lori Tribble	• Renamed to final
15.2.1		Lori Tribble	• Applied addenda e
15.2.2		Lori Tribble	• Applied addenda f
15.2.3		Lori Tribble	• Applied addenda g
15.2.4	11-Nov-2018	Lori Tribble	• Removed some highlights
15.2.final	11-Nov-2018	Lori Tribble	• Accepted all changes and updated revision
15.2.5	13-Dec-2018	Lori Tribble	• Reverted back to 15.2.4 due to formatting issues. Accepted all changes again and updated revision and date.
15.2.final2	14-Dec-2018	Lori Tribble	• Updated date and revision.
16.0.1	19-Aug-2019	Lori Tribble	• Converted to docx. Updated TOC.
16.0.2	19-Aug-2019	Lori Tribble	• Applied Addenda h.
16.0.3	26-Aug-2019	Lori Tribble	• Applied Addenda i
16.0.4	27-Sug-2019	Lori Tribble	• Applied Addenda j
16.0.5	28-Aug-2019	Lori Tribble	• Applied Addenda k
16.0.6	29-Aug-2019	Lori Tribble	• Applied Addenda l
16.0.7	29-Aug-2019	Lori Tribble	• Applied Addenda m
16.0.8	29-Aug-2019	Lori Tribble	• Applied Addenda n
16.0.9	30-Aug-2019	Lori Tribble	• Applied Addenda o
16.0.10	30-Aug-2019	Lori Tribble	• Applied Addenda p
16.0.11	30-Aug-2019	Lori Tribble	• Applied Addenda r
16.0.12	30-Aug-2019	Lori Tribble	• Applied Addenda s
16.0.13	31-Aug-2019	Lori Tribble	• Applied Errata
16.0.14	19-Sep-2019	Lori Tribble	• Applied all changes. Formatting changes.
16.0.15	25-Sep-2019	Lori Tribble	• Applied Errata.
16.0.final	25-Sep-2019	Lori Tribble	• Renamed to Final
16.0.final.v2	11-Nov-2019	Emily Hayes	• Renamed to Final.v2
16.1	10-Dec-2019	Lori Tribble	• Applied Errata, added PR21 and PR22 items, renamed to 16.1
18.0_v1	4-Oct-2020	Lori Tribble	• Updated to version 18.0, applied Add ai, applied Add aj, applied addenda bf, applied addenda bg, applied addenda bj, applied addenda fix1, applied addenda fix2, applied addenda misc1, applied addenda al, applied fix3, applied aq, applied as Applied misc3 and misc4 (under misc3 author)
18.0_v2	10-Oct-2020	Lori Tribble	• Applied addenda misc2
18.0_v3	11-Oct-2020	Lori Tribble	• Applied Errata
18_v4	13-Oct-2020	Lori Tribble	• Applied Misc5
18_v5	13-Oct-2020	Lori Tribble	• Applied BTLWG-264
18_v6	18-Oct-2020	Lori Tribble	• Applied BTLWG-1025 • Moved new Notes to Tester to above Test Steps. Left existing to be handled via errata in 135.1.

BACnet Testing Laboratories - Specified Tests

			<ul style="list-style-type: none"> • Fixed Error Code and Error Class to have quotes around them. • Fixed numbering of some tests.
18.0_v7	29-Oct-2020	Lori Tribble	• Final cleanup after voter comments.
18.1_v1	3-Jan-2021	Lori Tribble	• Creation of document from 18.0_v7
18.1_v4	25-Jan-2021	Lori Tribble	• Added tests from Interim document.
18.1_v5	8-Feb-2021	Lori Tribble	• Added missing WriteGroup tests from Interim document.
18.1_v6	13-Feb-2021	Lori Tribble	• Applied final vote comments.
20.0_v1	18-Nov-2021	Lori Tribble	• Applied fix addenda
20.0_v2	19-Nov-2021	Lori Tribble	• Applied alm addenda
20.0_v3	21-Nov-2021	Lori Tribble	• Applied bc addenda (no changes needed)
20.0_v4	11-Dec-2021	Lori Tribble	• Applied bi addenda
20.0_v5	11-Dec-2021	Lori Tribble	• Applied bk addenda
20.0_v6	11-Dec-2021	Lori Tribble	• Applied cov addenda
20.0_v7	12-Dec-2021	Lori Tribble	• Applied misc1 addenda (no changes)
20.0_v8	12-Dec-2021	Lori Tribble	• Applied log addenda
20.0_v9	15-Dec-2021	Lori tribble	• Applied PR13-PR18 addenda
20.0_v10	15-Dec-2021	Lori Tribble	• Applied bd addenda
20.0_v11	16-Dec-2021	Lori Tribble	• Applied Errata (Part 1)
20.0_v15	02-Jan-2022	Lori Tribble	• Applied comments from voters
20.0_v16	14-Jan-2022	Carl Neilson	• Errata
20.0.1_draft2	25-Apr-2022	Lori Tribble	• Applied crs1 to 20.0.1_draft1 provided by Carl
20.0.1_draft3	1-May-2022	Lori Tribble	• Applied Errata to 20.0.1 and updated TOC.
20.0.1_draft4	7-May-2022	Lori Tribble	• Applied Errata
20.0.1_draft5	19-May-2022	Lori Tribble	• Applied Errata BTLWG-1271
20.0.1_draft6	13-Jun-2022	Lori Tribble	• Applied comments from voting members
20.0.1_draft7	15-Jun-2022	Carl Neilson	• Applied TP validation fixes
20.0.1_final	16-Jun-2022	Emily Hayes	• Final
23.0_v1	25-Oct-2022	Lori Tribble	• Applied vote1 (af, bc, bl, br) changes.
23.0_v2	1-Nov-2022	Lori Tribble	• Applied vote2 (bq, bs, bt, bu, bz) changes.
23.0_v3	1-Nov-2022	Lori Tribble	• Applied vote3 (cd, improvements1) changes.
23.0_v4	4-Nov-2022	Lori Tribble	• Applied vote4 (cr1 and cr2) changes
23.0_v5	11-Nov-2022	Lori Tribble	• Applied Errata as of 11/11/2022
23.0_v6		Carl Neilson	• Validation process before formal vote.
23.0_v7	21-Dec-2022	Lori Tribble	• Applied comments from formal vote.
23.0.1_v1	24-Feb-2023	Lori Tribble	• Applied Errata to 23.0.final
23.1_v2	23-May-2023	Lori Tribble	• Applied cr1, fix, errata
23.1_v3	14-Jun-2023	Lori Tribble	• Corrected TOC entries for several tests
23.1_final	15-Jun-2023	Emily Hayes	• Final
23.2	21-Dec-2023	Michael Osborne	• Add 135.1-2023
23.3	12-Feb-2024	Michael Osborne	• Applied 23.1 fix addendum
26.0_v1	7-Oct-2024	Lori Tribble	• Initial version for 26.0 based on 23.3_v4 with fix1, fix2, fix3, fix4 and epics added.
26.0_v2	9-Oct-2024	Lori Tribble	• Applied cr2
26.0_v3	17-Oct-2024	Lori Tribble	• Applied bq, br, bv, ca, cc, cf
26.0_v4	20-Oct-2024	Lori Tribble	• Applied imp1 and imp2
26.0_v5	23-Oct-2024	Lori Tribble	• Applied imp3 and imp4
26.0_v6	23-Oct-2024	Lori Tribble	• Applied cr2

BACnet Testing Laboratories - Specified Tests

26.0 v7	24-Oct-2024	Michael Osborne	<ul style="list-style-type: none"> • Added errata • Removed extra returns • Formatted some tests
26.1 v1	3-Sep-2025	Lori Tribble	<ul style="list-style-type: none"> • Applied TP26.0 Fix1
26.1 v2	4-Sep-2025	Lori Tribble	<ul style="list-style-type: none"> • Applied TP26.0 Fix2
26.1 v3	24-Sep-2025	Lori Tribble	<ul style="list-style-type: none"> • Applied TP26.0 Fix3
26.1 v5	25-Sep-2025	Lori Tribble	<ul style="list-style-type: none"> • Applied TP26.0 cr1 (there is no v4 of this document).
26.1 v6	26-Sep-2025	Lori Tribble	<ul style="list-style-type: none"> • Applied TP26.0 impl
26.1 v7	22-Oct-2025	Lori Tribble	<ul style="list-style-type: none"> • Final Tweaks
26.1 draft2	8-Nov-2025	Michael Osborne	<ul style="list-style-type: none"> • Final, post-vote
26.1 final	19-Nov-2025	Emily Hayes	<ul style="list-style-type: none"> • Prep for publication