BACnet® TESTING LABORATORIES ADDENDA

Addendum bj to BTL Test Package 16.1

Revision 2
Revised 7/17/2020

Approved by the BTL Working Group on June 11, 2020.
Approved by the BTL Working Group Voting Members on July 3, 2020.
Published on July 22, 2020.

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

FOREWORD

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

In the following document, language to be added to existing clauses within the BTL Test Package 16.1 is indicated through the use of *italics*, while deletions are indicated by ~~strikethrough~~. Where entirely new subclauses are proposed to be added, plain type is used throughout

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

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

BTL-16.1bj-1: BACnet SC Data link Tests - BTLWG-788

Overview:

Add testing for the new BACnet SC Data Link Layer specified in Addenda bj to 135-2016.

Changes:

BTL Checklist Changes

[In BTL Checklist, replace Data Link Layer - Secure Connect section]

Support	Listing	Option
Data Link Layer - Secure Connect		
	R	Base Requirements
	C[1]	Is able to operate as a node without a local hub function
	C[1]	Is able to operate as a hub
	O	Supports direct connections
	O[2]	Is able to initiate direct connections
	O[2]	Is able to accept direct connections
[1] At least one of these options must be supported. [2] At least one of these options must be supported if the device supports direct connections.		

[In BTL Checklist, modify Secure Connection items in the Network Management section]

Network Management - Secure Connect Hub - B		
	~~R~~[+] *R*	Base Requirements
~~[+]Contact BTL for interim tests for this BIBB.~~		
Network Management - Secure Connect Direct Connect - A		
	~~R~~[+] *R*	Base Requirements
~~[+]Contact BTL for interim tests for this BIBB.~~		
Network Management - Secure Connect Direct Connect - B		
	~~R~~[+] *R*	Base Requirements
~~[+]Contact BTL for interim tests for this BIBB.~~		

BTL Test Plan Changes

[In BTL Test Plan, replace section 9.10 Data Link Layer - Secure Connect]

9.10 Data Link Layer - Secure Connect

9.10.1 Base Requirements

Base requirements must be met by any IUT that supports BACnet/Secure Connect.

BTL - 14.YY.1.1.1 - Connect and Maintain Hub Connection Test		
	Test Conditionality	Must be executed.

	Test Directives	Repeat with IUT configured with a hub URI that requires DNS resolution, and with a URI that does not.
	Testing Hints	
BTL - 14.YY.1.1.5 - Unicast Through Hub Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.6 - Unicast to Hub Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.7 - Local Broadcast Initiation Test		
	Test Conditionality	If the IUT never initiates broadcasts, this test shall be skipped.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.8 - Local Broadcast Execution Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.9 - VMAC Uniqueness Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.17 - Configurable Reconnect Timeout Test		
	Test Conditionality	If the IUT has a fixed reconnect timeout, this test shall be skipped.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.18 - Fixed Reconnect Timeout Test		
	Test Conditionality	If the IUT has a configurable reconnect timeout, this test shall be skipped.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.2.1 - Direct Connect Not Supported - NAK Address Resolution Test		
	Test Conditionality	If the IUT cannot be configured to refuse direct connections, this test shall be skipped.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.2.5 - Connect-Request Response Wait Time Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.2.6 - HTTP 1.1 Fallback Test		
	Test Conditionality	This test shall be executed if the IUT supports BACnet/SC over HTTP 2 or later.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.2.7 - Rejection of Invalid Certificate Outgoing Connection Test		
	Test Conditionality	Must be executed.
	Test Directives	Repeat with an expired certificate. Repeat with a certificate not signed by the locally configured CA.
	Testing Hints	
BTL - 14.YY.1.2.8 - No Additional Certificate Checks Performed Test on Outgoing Connections		
	Test Conditionality	Must be executed.
	Test Directives	Repeat with a certificate: - with a domain different from the IUT's domain; - with an invalid organization;

		- with a hostname that does not match the device that presents the certificate; - with a hostname which contains multiple periods in a row.
	Testing Hints	
BTL - 14.YY.1.2.9 - Invalid WebSocket Data Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	

9.10.2 Is Able to Operate as a Node Without a Local Hub Function

The IUT supports operating as a node without a local hub function enabled.

BTL - 14.YY.1.1.2 - Connect to Failover Hub Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.3 - Connect to Failover Hub on Startup Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.4 - Reconnect to Primary Hub Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.10 - UUID Persistence Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.11 - UUID Persistence When VMAC Changes Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.12 - Unknown 'Must Understand' is True Message Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.13 - Unknown 'Must Understand' is False Message Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.14 - Multiple Header Options Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.15 - Advertisement-Solicitation Execution Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.16 - Heartbeat-Request Initiation Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.2.2 - Malformed BVLC Test		
	Test Conditionality	Must be executed.
	Test Directives	Apply the test with the IUT configured as a basic node connected to the TD as the primary hub.
	Testing Hints	

BTL - 14.YY.1.2.3 - Discard BVLC with Wrong Address Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.2.4 - Hub Connector Ignores Malformed Hub URIs Test		
	Test Conditionality	Must be executed.
	Test Directives	Repeat with a generally invalid URI and with a URI using an invalid scheme.
	Testing Hints	
BTL - 14.YY.1.3.1 - Configuration Via PEM Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.3.2 - Configuration Tool Accepts Arbitrary Valid Certificate Parameters Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.3.3 - Factory Defaults Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	

9.10.3 Is Able to Operate as a Hub

The IUT is able to operate as a hub (contains a hub function).

BTL - 14.YY.2.1.1 - Local Broadcast Initiation Test		
	Test Conditionality	If the IUT never initiates broadcasts, this test shall be skipped.
	Test Directives	
	Testing Hints	
BTL - 14.YY.2.1.2 - Local Broadcast Execution Test		
	Test Conditionality	Must be executed.
	Test Directives	Repeat test with 1, and N clients attached to the hub, where N is any value supported by the IUT.
	Testing Hints	
BTL - 14.YY.2.1.3 - Minimum NPDU Forwarding Size Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.2.1.4 - Failover Hub Connects to Primary Hub Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.2.1.5 - Failover Hub's Local Node Connects to Failover Hub Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.2.1.6 - Failover hub Split Horizon Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.2.1.7 - Hub Forwards Unicast BVLCs Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.2.1.8 - No Additional Certificate Checks Performed Test On Incoming Connections		
	Test Conditionality	Must be executed.
	Test Directives	Repeat with a certificate:

		- with a domain different from the IUT's domain; - with an invalid organization; - with a hostname that does not match the device that presents the certificate; - with a hostname which contains multiple periods in a row.
	Testing Hints	
BTL - 14.YY.2.1.9 - Duplicate Connection Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.2.1.10 - Heartbeat-Request Execution Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.2.2 - Malformed BVLC Test		
	Test Conditionality	Must be executed.
	Test Directives	Apply the test with the IUT configured as the primary hub.
	Testing Hints	
BTL - 14.YY.2.2.1 - Hub Discards BVLCs with Non-connected VMAC Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.2.2.2 - Connect-Request Wait Time Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.2.2.3 - VMAC Collision Detection Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.2.2.4 - Rejection of Invalid Certificate Incoming Connection Test		
	Test Conditionality	Must be executed.
	Test Directives	Repeat with an expired certificate. Repeat with a certificate not signed by the locally configured CA.
	Testing Hints	

9.10.4 Supports Direct Connections

The IUT supports direct connections.

VERIFY Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims support for initiating and/or accepting direct connections.
	Testing Hints	
BTL - 14.YY.3.1.1.1 - Unicast Through Direct Connect Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.3.1.1.2 - Direct Connect Disconnect Test		
	Test Conditionality	If the IUT does not drop connections, this test shall be skipped.
	Test Directives	
	Testing Hints	
BTL - 14.YY.3.1.1.3 - Direct Connect Establishment Failover Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.3.1.2.1 - Discard Broadcast BVLC Received on Direct Connect Test		

	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.2.2 - Malformed BVLC Test		
	Test Conditionality	Must be executed.
	Test Directives	Apply the test with the IUT connected to the TD via a direct connection.
	Testing Hints	
BTL - 14.YY.1.2.3 - Discard BLVC with Wrong Address Test		
	Test Conditionality	Must be executed.
	Test Directives	Apply the test with the IUT connected to the TD via a direct connection.
	Testing Hints	

9.10.5 Is Able to Accept Direct Connections

The IUT supports accepting incoming direct connections.

VERIFY Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims support for direct connections.
	Testing Hints	
BTL - 14.YY.3.2.1.1 - Direct Connect Acceptance Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.3.2.1.2 - No Additional Certificate Checks Performed Test On Incoming Connections		
	Test Conditionality	Must be executed.
	Test Directives	Repeat with a certificate: - with a domain different from the IUT's domain; - with an invalid organization; - with a hostname that does not match the device that presents the certificate; - with a hostname which contains multiple periods in a row.
	Testing Hints	
BTL - 14.YY.3.2.2.1 - Connect-Request Wait Time Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.3.2.2.2 - Direct-Connect Duplicate Connection for IUT Accepted Connections Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.3.2.2.3 - Direct-Connect Duplicate Connection for IUT Initiated Connections Test		
	Test Conditionality	If the IUT cannot be configured to both support initiating and accepting direct connections, this test shall be skipped. If the IUT does not support 2 or more simultaneous direct connect WebSocket connections, this test shall be skipped.
	Test Directives	
	Testing Hints	
BTL - 14.YY.3.2.2.4 - VMAC Collision Detection Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.3.2.2.5 - Rejection of Invalid Certificate Incoming Connection Test		
	Test Conditionality	Must be executed.
	Test Directives	Repeat with an expired certificate. Repeat with a certificate not signed by the locally configured CA.
	Testing Hints	

9.10.6 Is Able to Initiate Direct Connections

The IUT supports initiating direct connections.

VERIFY Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims support for direct connections.
	Testing Hints	
BTL - 14.YY.3.3.1.1 - Direct Connect Establishment Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.3.3.1.2 - Direct Connect Multiple URI Test		
	Test Conditionality	If the IUT does not dynamically determine direct connect per URIs, this test shall be skipped.
	Test Directives	
	Testing Hints	
BTL - 14.YY.3.3.2.1 - Invalid Web Socket Scheme In Configured Direct Connect URI Test		
	Test Conditionality	If the IUT does not support configured peer direct connect URI, this test shall be skipped.
	Test Directives	
	Testing Hints	
BTL - 14.YY.3.3.2.2 - Invalid Web Socket Scheme In Discovered Direct Connect URI Test		
	Test Conditionality	If the IUT does not support discovering a peer's direct connect URI, this test shall be skipped.
	Test Directives	
	Testing Hints	
BTL - 14.YY.3.3.2.3 - Rejection of Invalid Certificate Outgoing Connection Test		
	Test Conditionality	Must be executed.
	Test Directives	Repeat with an expired certificate. Repeat with a certificate not signed by the locally configured CA.
	Testing Hints	

[In BTL Test Plan, add to section 10.1.1 Network Management - Routing - Base Requirements]

BTL - 10.2.X3 - Data Attributes Forwarding Test		
	Test Conditionality	If the IUT is not able to route between multiple networks which support data_attributes, this test shall be skipped.
	Test Directives	Execute the test with multiple attributes in the message to be routed.
	Testing Hints	As of PR 22, BACnet/SC is the only standard datalink which supports data_attributes
BTL - 10.2.X4 - Data Attributes Dropping Test		
	Test Conditionality	If the IUT is not able to route between a network which supports data_attributes and one which does not, this test shall be skipped.
	Test Directives	
	Testing Hints	As of PR 22, BACnet/SC is the only standard secure datalink.
BTL - 10.2.X5 - Secure Path Test		
	Test Conditionality	If the IUT is not able to route between multiple secure networks, this test shall be skipped.
	Test Directives	
	Testing Hints	As of PR 22, BACnet/SC is the only standard datalink which supports data_attributes
BTL - 10.2.X6 - Insecure Path Test		
	Test Conditionality	If the IUT is not able to route between a secure network and an insecure network, this test shall be skipped.
	Test Directives	
	Testing Hints	As of PR 22, BACnet/SC is the only standard secure datalink.

[In BTL Test Plan, replace sections 10.9, 10.10, 10.11]

10.9 Network Management - Secure Connect Hub - B

10.9.1 Base Requirements

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

VERIFY Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims the "Is able to operate as a hub" option under Data Link Layer - Secure Connect.
	Testing Hints	

10.10 Network Management - Secure Connect Direct Connect - A

10.10.1 Base Requirements

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

VERIFY Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims the "Is able to initiate direct connections" option under Data Link Layer - Secure Connect.
	Testing Hints	

10.11 Network Management - Secure Connect Direct Connect - B

10.11.1 Base Requirements

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

VERIFY Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims the "Is able to accept direct connections" option under Data Link Layer - Secure Connect.
	Testing Hints	

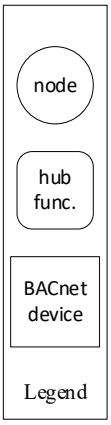
BTL Specified Tests Changes

[Insert new **Clause 14.YY**, p. 643]

14.YY Secure Connect Functionality Tests

This clause defines the tests necessary to demonstrate Secure Connect functionality, as defined in Annex YY of the BACnet Standard.

In the diagrams that follow, the following legend applies. Nodes and hub functions are shown within the BACnet device in which they reside by having the circle or rounded square located inside a BACnet device rectangle.



Secure Connect differs from other datalinks in that a single network consists of numerous logical connections instead of a shared bus. While the messages do usually exist on a shared ethernet segment, they are described as if each WebSocket is a separate link.

Where it is not clear which WebSocket the messages are expected on, the PORT keyword is used to identify the WebSocket. This construct is mostly used when the IUT has multiple connections such as when it is a hub or participating in direct connections.

14.YY.1 Basic Node Tests

This group of tests verifies secure connect devices operating in a non-hub mode. The logical configuration of the network used for these tests is shown in Figure X1. The test descriptions in this clause assume that the TD plays the role of all of the other devices in the network configuration. For the tests in this section, unless specified through a PORT parameter, messages specified by the test are on the IUT to TD hub WebSocket connection (primary or failover).

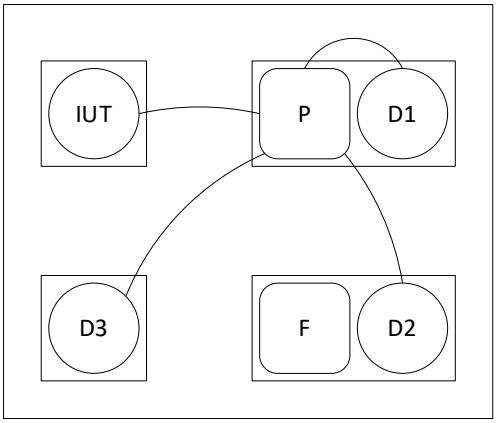


Figure X1: Network setup for basic node tests showing connections when the primary hub is active.

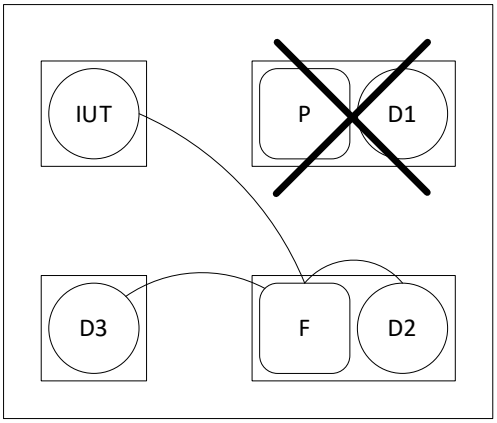


Figure X2: Network setup for basic node tests showing connections when the primary hub is inactive.

14.YY.1.1 Basic Node Positive Tests

14.YY.1.1.1 Connect and Maintain Hub Connection Test

Reference: YY.5.2

Purpose: To verify that the IUT connects to the configured primary hub and maintains the connection over time.

Test Concept: With the IUT configured to connect to the TD as the primary hub, allow the IUT to connect. Wait an arbitrary amount of time and have the hub silently close the WebSocket. Verify that the IUT detects the closure and attempts to reconnect to the hub. Allow the hub to accept the new connection. Wait an arbitrary amount of time and have the hub request a disconnect. Verify that the IUT acknowledges the disconnect. Verify that the IUT attempts to reconnect to the hub. Allow the hub to accept the new connection.

Configuration Requirements: The TD is configured as the primary hub, and the IUT's primary hub URI is configured to reference it.

Test Steps:

1. MAKE(the IUT connect to the primary hub)
2. WAIT a tester selected amount of time
3. MAKE(the hub close the WebSocket)
4. CHECK(that the IUT opens a new WebSocket with the primary hub)
5. RECEIVE Connect-Request,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)
6. TRANSMIT Connect-Accept,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)
7. WAIT a tester selected amount of time
8. TRANSMIT Disconnect-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
9. RECEIVE Disconnect-ACK,
   'Message ID' = M2,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- 'Data Options' absent
10. MAKE(the TD close the WebSocket)
11. CHECK(that the IUT opens a new WebSocket with the TD)
12. RECEIVE Connect-Request,
    'Message ID' = (M3: any valid value),
    -- 'Originating Virtual Address' absent
    -- 'Destination Virtual Address' absent
    'Destination Options' (absent or any valid value),
    -- '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. TRANSMIT Connect-Accept,
'Message ID' = M3,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' (absent or any valid value),
-- '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)

14.YY.1.1.2 Connect to Failover Hub Test

Reference: YY.5.2

Purpose: To verify that the IUT connects to the configured failover hub and maintains the connection when the connection to the primary is lost and reconnects to the primary when it returns.

Test Concept: With the IUT configured with a primary and a failover hub allow the IUT to connect to the primary. Wait an arbitrary amount of time and have the hub silently close the WebSocket and stop responding on that port. Verify that the IUT detects the closure and attempts to reconnect to the primary hub. Verify that after the reconnect attempt fails, the IUT attempts to connect to the failover hub. Allow the connection to succeed.

Configuration Requirements: The IUT configured to connect to the TD as the primary hub, and as the failover hub.

Test Steps:

1. MAKE(the IUT connect to the primary hub)
2. WAIT a tester selected amount of time
3. MAKE(the hub close the WebSocket)
4. CHECK(that the IUT opens a new WebSocket with the primary hub)
5. RECEIVE PORT (IUT-TD primary hub WebSocket)
Connect-Request,
'Message ID' = (M1: any valid value),
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' (absent or any valid value),
-- '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)
6. TRANSMIT PORT (IUT-TD primary hub WebSocket)
Connect-Accept,
'Message ID' = M1,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' (absent or any valid value),
-- '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)
7. WAIT a tester selected amount of time
8. MAKE(the TD, as primary hub, close the WebSocket connection to the IUT and refuse future connections)
9. CHECK(that the IUT attempts to open a new WebSocket with the the primary hub)
10. CHECK(that the IUT opens a WebSocket with the failover hub)
11. RECEIVE PORT (IUT-TD failover hub WebSocket)
Connect-Request,
'DA' = D2,

'Message ID' = (M2: any valid value),
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' (absent or any valid value),
-- '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)

6. TRANSMIT PORT (IUT-TD failover hub WebSocket)
   Connect-Accept,
   'SA' = D2,
   'Message ID' = M2,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- 'Data Options' absent
   'VMAC Address' = (D2 VMAC),
   'Device UUID' = (D2's UUID),
   'Maximum BVLC Length' = (the D2's maximum BVLC accepted length),
   'Maximum NPDU Length' = (the D2's maximum NPDU accepted length)

14.YY.1.1.3 Connect to Failover Hub on Startup Test

Reference: YY.5.2

Purpose: To verify that the IUT connects to the configured failover hub when it powers on and the primary is offline.

Test Concept: With the IUT configured with a primary and a failover hub, the primary hub offline, and the IUT powered off, power on the IUT. Verify that the IUT attempts to connect to the primary. Verify that the IUT attempts to connect to the failover hub when the attempt to the primary fails.

Configuration Requirements: The IUT configured to connect to the TD as the primary hub, and as the failover hub. The primary hub is offline. The IUT starts the test powered off.

Test Steps:

1. MAKE(power on the IUT)
2. CHECK(that the IUT attempts open a WebSocket to the primary hub)
3. CHECK(that the IUT opens a new WebSocket with the failover hub)
4. RECEIVE PORT (IUT-TD failover hub WebSocket)
   Connect-Request,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)
5. TRANSMIT PORT (IUT-TD failover hub WebSocket)
   Connect-Accept,
   SA = D2,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- 'Data Options' absent
   'VMAC Address' = (D2's VMAC),
   'Device UUID' = (D2's UUID),
   'Maximum BVLC Length' = (the D2's maximum BVLC accepted length),
   'Maximum NPDU Length' = (the D2's maximum NPDU accepted length)

14.YY.1.1.4 Reconnect to Primary Hub Test

Reference: YY.5.2, YY.6.1

Purpose: To verify that the IUT reconnects to the configured primary hub within 600 seconds of when it comes back online.

Test Concept: The test starts with the IUT connected to the failover hub and the primary hub offline. Wait an arbitrary amount of time and have the primary hub come back online. Verify that the IUT attempts to reconnects to the primary within 600 seconds of it coming back online.

Configuration Requirements: The IUT configured to connect to the TD as the primary hub, and as the failover hub. The test starts with the primary hub offline and the IUT connected to the failover hub.

Test Steps:

1. WAIT a tester selected amount of time
2. MAKE(bring the primary hub online)
4. CHECK(that the IUT opens a new WebSocket with the primary hub within 600 seconds)
5. RECEIVE PORT (IUT-TD primary hub WebSocket)
   Connect-Request,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)
6. TRANSMIT PORT (IUT-TD primary hub WebSocket)
   Connect-Accept,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)

14.YY.1.1.5 Unicast Through Hub Test

Reference: YY2.5, YY.5.4

Purpose: Verify that the IUT correctly composes unicasts aimed at a device through the hub.

Test Concept: With the IUT connected to the BACnet/SC network, send a ReadProperty from OD to the IUT. Verify that the IUT accepts the request and responds with a ReadProperty-ACK to OD, the ack properly composed and sent to the hub.

Configuration Requirements: The TD is configured as the primary hub and the IUT is connected to it. OD is any device connected to the BACnet/SC network through the primary hub (it could be TD’s node, or another device).

Test Steps:

1. TRANSMIT Encapsulated-NPDU,
   'Message ID' = (M1: any valid value),
   'Originating Virtual Address' = (OD's VMAC),
   'Destination Options' = (absent or a valid list of options),
   'Data Options' = (X'01') -- Secure Path
   'BACnet NPDU' =
   ReadProperty-Request
   'Object Identifier' = (O: any object in the IUT),
   'Property Identifier' = Object_Name
2. RECEIVE Encapsulated-NPDU,
   'Message ID' = (M2: any valid value),

-- 'Originating Virtual Address' absent
'Destination Virtual Address' = (OD's VMAC),
'Destination Options' = (absent or a valid list of options),
'Data Options' = ((X'01') or a list of data options including Secure Path),
'BACnet NPDU' =
ReadProperty-ACK
'Object Identifier' = (O),
'Property Identifier' = Object_Name,
'Property Value' = (the value from the EPICS)

14.YY.1.1.6 Unicast to Hub Test

Reference: YY.5.4

Purpose: Verify that the IUT correctly composes unicasts aimed at the hub.

Test Concept: With the IUT connected to the BACnet/SC network, send a ReadProperty from the node on the same device as the primary hub to the IUT. Verify that the IUT accepts the request and responds with a ReadProperty-ACK, the ack properly composed and sent to the hub.

Configuration Requirements: The TD is configured as the primary hub.

Test Steps:

1. TRANSMIT PORT (IUT-TD primary hub WebSocket)
   Encapsulated-NPDU,
   'Message ID' = (M1: any valid value),
   'Originating Virtual Address' = TD's VMAC,
   -- 'Destination Virtual Address' absent
   'Destination Options' = (absent or any valid value),
   'Data Options' = (X'01'),
   'BACnet NPDU' =
   ReadProperty-Request,
   'Object Identifier' = (the IUT's Device object),
   'Property Identifier' = Object_Name
2. RECEIVE PORT (IUT-TD primary hub WebSocket)
   Encapsulated-NPDU,
   'Message ID' = M2,
   -- 'Originating Virtual Address' absent
   'Destination Virtual Address' = TD's VMAC,
   'Destination Options' (absent or any valid value),
   'Data Options' = ((X'01') or a list of valid header options including Secur Path),
   'BACnet NPDU' =
   ReadProperty-ACK,
   'Object Identifier' = (the IUT's Device object),
   'Property Identifier' = Object_Name,
   'Property Value' = (the IUT's device object name)

14.YY.1.1.7 Local Broadcast Initiation Test

Reference: YY.3.1.2

Purpose: To verify that broadcasts are sent to the hub, and are sent with the Local broadcast VMAC address

Test Concept: Make the IUT generate a broadcast. Verify that the broadcast is correctly formed and sent to the primary hub.

Configuration Requirements: The TD is configured as the primary hub and the IUT is connected to it.

Test Steps:

1. MAKE(the IUT generate a broadcast)
2. RECEIVE Encapsulated-NPDU,
   'Message ID' = (M1: any valid value)
   'Originating Virtual Address' absent
   'Destination Virtual Address' = (Local Broadcast VMAC, X'FFFFFFFFFFFF')
   'Destination Options' = (absent or a valid list of options),
   'Data Options' = ((X'01') or a list of data options including Secure Path)

'BACnet NPDU' = (any valid broadcastable NPDU)

14.YY.1.1.8 Local Broadcast Execution Test

Purpose: To verify that IUT correctly accepts and processes broadcast messages.

Test Concept: Send a broadcast WhoIs with no device range. Verify that the IUT sends an IAm in response through the primary hub.

Configuration Requirements: The TD is configured as the primary and the IUT is connected to it.

Test Steps:

1. TRANSMIT Encapsulated-NPDU,
   'Message ID' = (M1: any valid value),
   'Originating Virtual Address' = (OD's VMAC),
   'Destination Virtual Address' = (Local Broadcast VMAC, X'FFFFFFFFFFFF')
   'Destination Options' = (absent or a valid list of options),
   'Data Options' = (X'01') -- Secure Path
   'BACnet NPDU' =
   Who-Is-Request,
   'Device Instance Range Low Limit' = (IUT's device instance),
   'Device Instance Range High Limit' = (IUT's device instance)
2. RECEIVE Encapsulated-NPDU,
   'Message ID' = (M2: any valid value)
   'Originating Virtual Address' absent
   'Destination Virtual Address' = (Local Broadcast VMAC, X'FFFFFFFFFFFF')
   'Destination Options' = (absent or a valid list of options),
   'Data Options' = ((X'01') or a list of data options including Secure Path)
   'BACnet NPDU' =
   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)

14.YY.1.1.9 VMAC Uniqueness Test

Reference: H.7.X, YY.6.2.2

Purpose: To verify that the IUT will attempt to resolve its own VMAC and will generate a new Random-48 VMAC if a conflict is detected.

Test Concept: Have the IUT connect to the primary hub. During the connect sequence, the hub returns an NAK with an error code of NODE_DUPLICATE_VMAC. Verify that the IUT retries the connection with a Random-48 VMAC which differs from the initial connect attempt. Verify that the selected VMAC is valid. Again, the hub NAKs the connection request with an error code of NODE_DUPLICATE_VMAC. Verify that the IUT retries again with a valid Random-48 VMAC not equal to either of the previous VMACs.

Test Steps:

1. MAKE(the IUT initiate a hub connection TD)
2. CHECK(that the IUT attempts to open a new WebSocket with the TD)
3. RECEIVE Connect-Request,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- 'Data Options' absent
   'VMAC Address' = (VMAC1: 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. TRANSMIT BVLC-Result,
   'Message ID' = M1,

-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' (absent or any valid value),
-- 'Data Options' absent
'Result for BVLC Function' = X'06', -- Connect-Request
'Result Code' = X'01', -- NAK
'Error Header Marker' = X'00', -- not a header option problem
'Error Class' = COMMUNICATION,
'Error Code' = NODE_DUPLICATE_VMAC

4. RECEIVE Connect-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
'VMAC Address' = (VMAC2: IUT's new VMAC, different than VMAC1),
'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)
5. TRANSMIT BVLC-Result,
'Message ID' = M2,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' (absent or any valid value),
-- 'Data Options' absent
'Result for BVLC Function' = X'06', -- Connect-Request
'Result Code' = X'01', -- NAK
'Error Header Marker' = X'00', -- not a header option problem
'Error Class' = COMMUNICATION,
'Error Code' = NODE_DUPLICATE_VMAC
6. RECEIVE Connect-Request,
'Message ID' = (M3: any valid value),
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' (absent or any valid value),
-- 'Data Options' absent
'VMAC Address' = (VMAC3: IUT's new VMAC, different than VMAC1 & VMAC2),
'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. TRANSMIT Connect-Accept,
'Message ID' = M3,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' (absent or any valid value),
-- '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)

14.YY.1.1.10 UUID Persistence Test

Reference: YY.1.5.3

Purpose: To verify that the IUT contains a UUID and that it persists across resets

Test Concept: Configure the IUT and have it connect to the primary hub. Power off the IUT, wait 1 minute, and power on the IUT allowing it to connect again to the hub. Verify that the UUID in both connect messages is the same.

1. MAKE(IUT connect to the primary hub)
2. RECEIVE Connect-Request,
'Message ID' = (M1: any valid value),

-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' = (absent or a valid list of options),
-- 'Data Options' absent
'VMAC Address' = (IUT's VMAC),
'Device UUID' = (U: any valid UUID),
'Maximum BVLC Length' = (the IUT's maximum BVLC accepted length),
'Maximum NPDU Length' = (the IUT's maximum NPDU accepted length)

3. TRANSMIT Connect-Accept,
'Message ID' = M1,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' = (absent or a valid list of options),
-- '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)
4. MAKE(power off the IUT)
5. WAIT 1 minute
6. MAKE(power on the IUT)
7. BEFORE the maximum time it takes for the IUT to reboot and re-establish a connection to the network
RECEIVE Connect-Request,
'Message ID' = (M2: any valid value),
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' = (absent or a valid list of options),
-- 'Data Options' absent
'VMAC Address' = (IUT's VMAC),
'Device UUID' = U,
'Maximum BVLC Length' = (the IUT's maximum BVLC accepted length),
'Maximum NPDU Length' = (the IUT's maximum NPDU accepted length)
8. TRANSMIT Connect-Accept,
'Message ID' = M2,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' = (absent or a valid list of options),
-- '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)

14.YY.1.1.11 UUID Persistence When VMAC Changes Test

Reference: YY.1.5.3

Purpose: To verify that the IUT's UUID is unrelated to its VMAC.

Test Concept: Configure the IUT and connect it to the network. Let it verify its VMAC successfully. Reset the IUT and when it connects to the network, indicate a VMAC conflict. After the IUT allocates a new VMAC, and send a new advertisement, verify that the UUID has not changed.

-- allow the IUT to connect and use its chosen VMAC
1. MAKE(the IUT connect to the primary hub)
2. RECEIVE Connect-Request,
'Message ID' = (M1: any valid value),
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' = (absent or a valid list of options),
-- 'Data Options' absent
'VMAC Address' = (V1: any valid value),

'Device UUID' = U,
'Maximum BVLC Length' = (the IUT's maximum BVLC accepted length),
'Maximum NPDU Length' = (the IUT's maximum NPDU accepted length)

3. TRANSMIT Connect-Accept,
'Message ID' = M1,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' = (absent or a valid list of options),
-- '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)

-- TD pretending a new device has joined using VMAC V1 with UUID U2
-- the TD will now reject the IUT's VMAC as if another device has connected with the IUT's VMAC

4. MAKE(the IUT reset)
5. BEFORE the maximum time it takes for the IUT to reboot and re-establish a connection to the netwrok
RECEIVE Connect-Request,
'Message ID' = (M2: any valid value),
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' = (absent or a valid list of options),
-- 'Data Options' absent
'VMAC Address' = V1,
'Device UUID' = U,
'Maximum BVLC Length' = (the IUT's maximum BVLC accepted length),
'Maximum NPDU Length' = (the IUT's maximum NPDU accepted length)
6. TRANSMIT BVLC-Result,
'Message ID' = M1,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' = (absent or a valid list of options),
-- 'Data Options' absent
'Result for BVLC Function' = X'06', -- Connect-Request
'Result Code' = X'01', -- NAK
'Error Header Marker' = X'00', -- not related to a header option
'Error Class' = COMMUNICATION,
'Error Code' = NODE_DUPLICATE_VMAC

-- verify that the IUT retries with a new VMAC and the same UUID

7. RECEIVE Connect-Request,
'Message ID' = (M2: any valid value),
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' = (absent or a valid list of options),
-- 'Data Options' absent
'VMAC Address' = (V2: any valid value different than V1),
'Device UUID' = U,
'Maximum BVLC Length' = (the IUT's maximum BVLC accepted length),
'Maximum NPDU Length' = (the IUT's maximum NPDU accepted length)
8. TRANSMIT Connect-Accept,
'Message ID' = M1,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' = (absent or a valid list of options),
-- '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)

14.YY.1.1.12 Unknown 'Must Understand' is True Message Test

Reference: YY.2.3, YY.3.1.4

Purpose: To verify that a unicast message is not ignored if the 'Must Understand' flag is set and the option is unknown or not supported.

Test Concept: With the IUT connected to the BACnet/SC network, send a ReadProperty request with a set of header options with one that the IUT does not know marked as 'Must Understand'. Verify that the IUT responds with a NAK and error class of COMMUNICATION and an error code of HEADER_NOT_UNDERSTOOD correctly identifying the option marked as 'Must Understand'. Verify that the IUT does not send a ReadProperty-ACK. Repeat with a globally broadcast Who-Is request.

Test Steps:

1. TRANSMIT Encapsulated-NPDU,
   'Message ID' = (M1: any valid value),
   'Originating Virtual Address' = (OVA: any valid value, including absent),
   'Destination Virtual Address' absent
   'Destination Options' = (X'BF0000000012',--proprietary,more,not M.U.,len=0,ven=0, opt=x12
   X'FF0000000000', --proprietary, more,M.U.,len=0,ven=0, opt=x00
   X'3F0000000034'--proprietary,not more,not M.U.,len=0,ven=0,opt=x34
)
   'Data Options' = (X'01') --secure path
   'BACnet NPDU' =
   ReadProperty-Request,
   'Object Identifier' = (the IUT's Device object),
   'Property Identifier' = Object_Name
2. RECEIVE BVLC-Result,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' = (OVA),
   'Destination Options' = (absent or a valid list of options),
   -- 'Data Options' absent
   'Result for BVLC Function' = X'01', -- Encapsulated-NPDU
   'Result Code' = X'01', -- NAK
   'Error Header Marker' = X'02', -- second header option
   'Error Class' = COMMUNICATION,
   'Error Code' = HEADER_NOT_UNDERSTOOD
3. CHECK(that the IUT does not send a ReadProperty-ACK)
4. TRANSMIT Encapsulated-NPDU,
   'Message ID' = (M2: any valid value),
   'Originating Virtual Address' = (OVA: any valid value, including absent),
   'Destination Virtual Address' = (X'FFFFFFFF', the local broadcast VMAC),
   'Destination Options' = (X'BF0000000012',--proprietary,more,not M.U.,len=0,ven=0, opt=x12
   X'FF0000000000', --proprietary, more,M.U.,len=0,ven=0, opt=x00
   X'3F0000000034'),--proprietary,not more,not M.U.,len=0,ven=0,opt=x34
   'Data Options' = (X'01'), --SecurePath
   'BACnet NPDU' =
   WhoIs-Request
5. CHECK(that the IUT does not route the message, send a BVLC-Result nor send an IAm-Request)

14.YY.1.1.13 Unknown 'Must Understand' is False Message Test

Reference YY.3.1.4

Purpose: To verify that a message is correctly processed when an unknown destination option is present whose 'Must Understand' flag is cleared.

Test Concept: With the IUT connected to the BACnet/SC network, send a ReadProperty request with a header option that the IUT does not know which is not marked as 'Must Understand'. Verify that the IUT accepts the request and responds with a ReadProperty-ACK.

Test Steps:

-- Unicast test
1. TRANSMIT Encapsulated-NPDU,
'Message ID' = (M1: any valid value),
'Originating Virtual Address' = (OVA: any valid value, including absent),
'Destination Options' = (X'3F0000000034' --proprietary,
-- not more,not M.U.,len=0,ven=0,opt=x34
)
'Data Options' = (X'01') -- secure path
'BACnet NPDU' =
ReadProperty-Request,
'Object Identifier' = (the IUT's Device object),
'Property Identifier' = Object_Name
2. RECEIVE Encapsulated-NPDU,
'Message ID' = (M2: any valid value),
-- 'Originating Virtual Address' absent
'Destination Virtual Address' = OVA,
'Destination Options' = (absent or a valid list of header options),
'Data Options' = ((X'01') 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)

-- Broadcast test
3. TRANSMIT Encapsulated-NPDU,
'Message ID' = (M3: any valid value),
'Originating Virtual Address' = (OVA: any valid value, including absent),
'Destination Virtual Address' = (X'FFFFFFFF' the local broadcast VMAC),
'Destination Options' = (X'3F0000000034'),--proprietary,not more,not M.U., len=0, ven=0,
-- opt=x34
'Data Options' = (X'01'), --SecurePath
'BACnet NPDU' =
WhoIs-Request
4. RECEIVE Encapsulated-NPDU,
'Message ID' = (M4: any valid value),
-- 'Originating Virtual Address' absent
'Destination Virtual Address' = (OVA, or the local broadcast VMAC),
'Destination Options' = (absent or a valid list of header options),
'Data Options' = ((X'01') or a list of valid header options including Secure Path),
'BACnet NPDU' =
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)

14.YY.1.1.14 Multiple Header Options Test

Reference: YY.2.3

Purpose: To verify that the IUT accepts and processes messages with varying numbers of header options.

Test Concept: With the IUT connected to the BACnet/SC network, send a ReadProperty request with more than 1 header options, with none marked as 'Must Understand'. Verify that the IUT responds with a ReadProperty-ACK.

1. 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'81', --secure path, more options

```
                                        X'BF0000000012', --proprietary,more,not M.U.,len=0,ven=0, opt=12
                                        X'3F0000000034'   --proprietary,not more,not M.U.,len=0,ven=0,
                                                --opt=x34
                                        )
        'BACnet NPDU' =
            ReadProperty-Request,
            'Object Identifier' =       (the IUT's Device object),
            'Property Identifier' =     Object_Name
2.  RECEIVE Encapsulated-NPDU,
        'Message ID' =                  (M2: any valid value),
        'Originating Virtual Address' absent
        'Destination Virtual Address' = (OVA: any valid value, including absent),
        'Destination Options'           (absent or any valid value),
        'Data Options' =                ( (X'01') or a list of valid header options including Secur Path),
        'BACnet NPDU' =
            ReadProperty-ACK,
            'Object Identifier' =       (the IUT's Device object),
            'Property Identifier' =     Object_Name,
            'Property Value' =          (the IUT's device object name)
```

14.YY.1.1.15 Advertisement-Solicitation Execution Test

Purpose: To verify that when executing an Advertisement-Solicitation, the IUT correctly initiates Advertisement messages.

Test Concept: With the IUT connected to the primary hub, send an Advertisement-Solicitation to the IUT and verify it responds with a correct Advertisement. Disconnect the primary hub from the network and wait for the IUT to connect to the failover hub. Send an Advertisement-Solicitation to the IUT and verify it responds with a correct Advertisement.

Test Steps:

```
1.  TRANSMIT Advertisement-Solicitation,
        'Message ID' =                  (M1: any valid value),
        'Originating Virtual Address' = (OVA: absent, or any node on the network),
        -- 'Destination Virtual Address' absent
        -- 'Destination Options' absent
        -- 'Data Options' absent
2.  RECEIVE Advertisement,
        'Message ID' =                  (M2: any valid value),
        -- 'Originating Virtual Address' absent
        'Destination Virtual Address' = (OVA or local broadcast VMAC),
        'Destination Options' =         (absent or a valid list of options),
        -- 'Data Options' absent
        'Result for BVLC Function' =    X'05',    -- Advertisement-Solicitation
        'Hub Connection Status' =       X'01',    -- connected to primary
        'Accept Direct Connections' =   (as per the IUT's configuration),
        'Maximum BVLC Length' =         (as per the IUT's configuration),
        'Maximum NPDU Length' =         (as per the IUT's configuration)
3.  MAKE(the primary hub go offline)
4.  WAIT until the IUT connects to the failover hub
5.  TRANSMIT Advertisement-Solicitation,
        'Message ID' =                  (M3: any valid value),
        'Originating Virtual Address' = (OVA: absent, or any node on the network),
        -- 'Destination Virtual Address' absent
        -- 'Destination Options' absent
        -- 'Data Options' absent
6.  RECEIVE Advertisement,
        'Message ID' =                  (M4: any valid value),
        -- 'Originating Virtual Address' absent
        'Destination Virtual Address' = (OVA or local broadcast VMAC),
        'Destination Options' =         (absent or a valid list of options),
        -- 'Data Options' absent
        'Result for BVLC Function' =    X'05',    -- Advertisement-Solicitation
```

'Hub Connection Status' = X'02', -- connected to failover
'Accept Direct Connections' = (as per the IUT's configuration),
'Maximum BVLC Length' = (as per the IUT's configuration),
'Maximum NPDU Length' = (as per the IUT's configuration)

14.YY.1.1.16 Heartbeat-Request Initiation Test

Reference: YY.2.14, YY.2.15, YY.6.3

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

Test Concept: With the IUT connected to the BACnet/SC network, send a ReadProperty request to the IUT every heartbeat interval / 2 seconds. Verify that the IUT does not initiate a Heartbeat-Request. Stop sending messages to the IUT. 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.

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) {
   TRANSMIT Encapsulated-NPDU,
   'Message ID' = (M: 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'01'),
   'BACnet NPDU' =
   ReadProperty-Request,
   'Object Identifier' = (the IUT's Device object),
   'Property Identifier' = Object_Name
   RECEIVE Encapsulated-NPDU,
   'Message ID' = M,
   'Originating Virtual Address' absent
   'Destination Virtual Address' = OVA,
   'Destination Options' (absent or any valid value),
   'Data Options' = ((X'01') or a list of valid header options including Secur Path),
   'BACnet NPDU' =
   ReadProperty-ACK,
   'Object Identifier' = (the IUT's Device object),
   'Property Identifier' = Object_Name,
   'Property Value' = (the IUT's device object name)
   WAIT ½ of IUT's heartbeat interval
   }
2. CHECK(that the IUT did not send a HeartBeat during step 1)

-- Since we already waited ½ of an heartbeat interval, only ½ of that interval is now given for the IUT to
-- generate a Heartbeat-Request
3. BEFORE ½ of IUT's heartbeat interval + 10s
   RECEIVE Heartbeat-Request,
   'Message ID' = M2,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' = (absent or any valid value),
   -- 'Data Options' absent
4. TRANSMIT Heartbeat-ACK,
   'Message ID' = M2,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' = (absent or any valid value),
   -- 'Data Options' absent

14.YY.1.1.17 Configurable Reconnect Timeout Test

Reference: YY.6.1

Purpose: To verify that a device adheres to its configurable reconnect timeout.

Test Concept: Turn on the IUT. When the IUT attempts to connect to the primary hub, the primary hub does not respond. Verify that the IUT waits at least the configured reconnect timeout, and no longer than 600 seconds before attempting to reconnect.

Configuration Requirements: The IUT is configured with the TD as the primary hub with no failover hub or as direct connection initiation peer of the TD. The IUT is configured with a tester selected reconnect timeout, RT, within the range supported by the IUT and within 2 .. 300 seconds. The IUT starts the test powered off. If the IUT has a fixed reconnect timeout, this test shall be skipped.

Test Steps:

1. MAKE(the IUT connect to the TD)
2. CHECK(that the IUT attempts to open a new WebSocket with the TD)
3. MAKE(place the TD in a mode where it will accept incoming connections)
4. WAIT RT seconds
5. BEFORE 600 - RT seconds
   RECEIVE PORT (IUT-TD primary hub WebSocket)
   Connect-Request,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)
6. TRANSMIT PORT (IUT-TD primary hub WebSocket)
   Connect-Accept,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)

14.YY.1.1.18 Fixed Reconnect Timeout Test

Reference: YY.6.1

Purpose: To verify that a device's fixed reconnect timeout is in the range 10 .. 30 seconds.

Test Concept: Turn on the IUT. When the IUT attempts to connect to the TD as the primary hub or as a direct connection peer of the TD, the TD does not respond. Verify that the IUT does not attempt to connect within 10 seconds. Then verify that the IUT does attempt to connect within the following 590 seconds.

Configuration Requirements: The IUT is configured with the TD as the primary hub with no failover hub, or as a direct connection peer of the TD. The IUT starts the test powered off. If the IUT does not have a fixed reconnect timeout this test shall be skipped.

Test Steps:

1. MAKE(the IUT connect to the TD)
2. CHECK(that the IUT attempts to open a new WebSocket with the TD)
3. MAKE(place the TD in a mode where it will accept incoming connections)

4. WAIT 10 seconds
5. BEFORE 590 seconds
   RECEIVE PORT (IUT-TD primary hub WebSocket)
   Connect-Request,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)
6. TRANSMIT PORT (IUT-TD primary hub WebSocket)
   Connect-Accept,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)

14.YY.1.2 Basic Node Negative Tests

14.YY.1.2.1 Direct Connect Not Supported - NAK Address Resolution Test

Reference YY.3.3

Purpose: To verify that devices configured to not accept direct connect requests will NAK an Address-Resolution request.

Test Concept: With the IUT configured to not accept direct connections, and with it connected to the BACnet/SC network, have device D3 send an Address-Resolution message to the IUT. Verify that the IUT NAKs the request.

Configuration Requirements: The IUT is configured to not accept direct connections.

Test Steps:

1. TRANSMIT Address-Resolution,
   'Message ID' = (M1: any valid value)
   'Originating Virtual Address' = D3,
   -- 'Destination Virtual Address' absent
   'Destination Options' = (absent or a valid list of options),
   -- 'Data Options' absent
2. RECEIVE BVLC-Result,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   'Destination Virtual Address' = D3
   'Destination Options' = (absent or a valid list of options),
   -- 'Data Options' absent
   'Result for BVLC Function' = X'02', -- Address-Resolution
   'Result Code' = X'01', -- NAK
   'Error Header Marker' = X'00', -- not a header option problem
   'Error Class' = COMMUNICATION,
   'Error Code' = OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED

14.YY.1.2.2 Malformed BVLC Test

Reference YY.3.1.5

Purpose: Verify that device NAKs malformed / unknown unicast BVLC and ignores malformed / unknown broadcast BVLC.

Test Concept: With the IUT connected to the BACnet/SC network, send a sequence of malformed unicast and broadcast BVLCs to the IUT. Verify that the IUT responds with an appropriate NAK to each unicast one and does not process nor route the messages.

Configuration Requirements: The IUT is connected to the BACnet/SC network as a node or hub.

Test Steps:

-- Invalid BVLC function
1. TRANSMIT
   'BVLC Function' = (IV: an invalid 1-octet value),
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   -- 'Destination Options' absent
   -- 'Data Options' absent
2. RECEIVE BVLC-Result,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' = (absent or a valid list of options),
   -- 'Data Options' absent
   'Result for BVLC Function' = IV, -- the supplied invalid BVLC function from the request
   'Result Code' = X'01', -- NAK
   'Error Header Marker' = X'00', -- not a header option problem
   'Error Class' = COMMUNICATION,
   'Error Code' = BVLC_FUNCTION_UNKNOWN
3. CHECK(that the IUT did not process nor forward the request)

-- Inclusion of an Originating Virtual Address when it is required to be absent
4. TRANSMIT Disconnect-Request,
   'Message ID' = (M2: any valid value),
   'Originating Virtual Address' = D3,
   -- 'Destination Virtual Address' absent
   'Destination Options' = (absent or a valid list of options),
   -- 'Data Options' absent
5. RECEIVE BVLC-Result,
   'Message ID' = M2,
   -- 'Originating Virtual Address' absent
   'Destination Virtual Address' = D3
   'Destination Options' = (absent or a valid list of options),
   -- 'Data Options' absent
   'Result for BVLC Function' = X'08', -- Disconnect-Request
   'Result Code' = X'01', -- NAK
   'Error Header Marker' = X'00', -- not a header option problem
   'Error Class' = (absent or COMMUNICATION),
   'Error Code' = (absent or OTHER)
6. CHECK(that the IUT did not process the request)

-- Inclusion of a 'Destination Virtual Address when it is required to be absent
7. TRANSMIT Disconnect-Request,
   'Message ID' = (M3: any valid value),
   -- 'Originating Virtual Address' absent,
   'Destination Virtual Address' = (IUT's VMAC),
   'Destination Options' = (absent or a valid list of options),
   -- 'Data Options' absent
8. RECEIVE BVLC-Result,
   'Message ID' = M3,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' = (absent or a valid list of options),
   -- 'Data Options' absent

'Result for BVLC Function' = X'08', -- Disconnect-Request
'Result Code' = X'01', -- NAK
'Error Header Marker' = X'00', -- not a header option problem
'Error Class' = (absent or COMMUNICATION),
'Error Code' = (absent or OTHER)

9. CHECK(that the IUT did not process the request)

-- A truncated message

10. TRANSMIT Encapsulated-NPDU,
'Message ID' = (M4: any valid value),
'Originating Virtual Address' = (OVA: absent, or D3 if IUT is configured as a hub),
'Destination Virtual Address' = (IUT's VMAC),
-- 'Destination Options' absent
-- 'Data Options' absent
-- no NPDU included in the message
11. RECEIVE BVLC-Result,
'Message ID' = M4,
-- 'Originating Virtual Address' absent
'Destination Virtual Address' = OVA,
'Destination Options' = (absent or a valid list of options),
-- 'Data Options' absent
'Result for BVLC Function' = X'01', -- Encapsulated-NPDU
'Result Code' = X'01', -- NAK
'Error Header Marker' = X'00', -- not a header option problem
'Error Class' = COMMUNICATION,
'Error Code' = MESSAGE_INCOMPLETE
12. CHECK(that the IUT did not process the request)

-- A message with extra octets added on

13. TRANSMIT Disconnect-Request,
'Message ID' = (M5: any valid value),
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
-- 'Destination Options' absent
-- 'Data Options' absent
(extra octets) = ({ X'81', --a bunch of octets that look like valid data options.
X'BF0000000012'),
X'3F0000000034')
14. RECEIVE BVLC-Result,
'Message ID' = M5,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' = (absent or a valid list of options),
-- 'Data Options' absent
'Result for BVLC Function' = X'08', -- Disconnect-Request
'Result Code' = X'01', -- NAK
'Error Header Marker' = X'00', -- not a header option problem
'Error Class' = COMMUNICATION,
'Error Code' = UNEXPECTED_DATA
15. CHECK(that the IUT did not process the request)

-- A truncated broadcast message

16. TRANSMIT Encapsulated-NPDU,
DESTINATION = GLOBAL_BROADCAST,
'Message ID' = (M6: any valid value),
-- 'Originating Virtual Address' absent,
'Destination Virtual Address' = (local broadcast VMAC),
'Destination Options' = ((X'01') -- secure path, more follows (but none are present)
)
-- 'Data Options' absent

17. CHECK(that the IUT does not send a BVLC-Result, did not process the message, nor route the message)

14.YY.1.2.3 Discard BVLC with Wrong Address Test

Reference: YY.4.2.2

Purpose: To verify that BVLCs with an incorrect VMAC are dropped.

Test Concept: With the IUT connected to the BACnet/SC network, send a ReadProperty to the IUT in an invalid BVLC message with a BVLC Destination Address that does not match the IUT. Verify that the IUT does not respond with a BVLC NAK nor with a ReadProperty-ACK.

Test Steps:

1. TRANSMIT Encapsulated-NPDU,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' (absent or any valid VMAC)
   'Destination Virtual Address' = (any non-broadcast VMAC other than the IUT's),
   -- 'Destination Options' absent
   'Data Options' = (X'01') -- Secure Path
   'BACnet NPDU' =
   ReadProperty-Request,
   'Object Identifier' = (O: any object in the IUT),
   'Property Identifier' = Object_Name
2. CHECK(that the IUT does not response with a BVLC-Result nor a ReadProperty-ACK)

14.YY.1.2.4 Hub Connector Ignores Malformed Hub URIs Test

Reference: YY.5.4

Purpose: Verify that the IUT's hub connector will not attempt to connect to malformed URIs or URIs with a scheme other than wss.

Test Concept: Configure the IUT with a primary hub URI with an invalid scheme or otherwise invalid URI. Verify that the IUT does not attempt to connect to the invalid URI.

Configuration Requirements: If the IUT cannot be configured with an invalid hub URI this test shall be skipped.

Test Steps:

1. MAKE(configure the primary hub URI in the IUT with a URI having a scheme other than wss)
2. CHECK(that the IUT does not attempt to connect to the URI)

14.YY.1.2.5 Connect-Request Response Wait Time Test

Reference: YY.6.2, YY.7.5.5

Purpose: To verify that the IUT will close the WebSocket if a response to a Connect-Request is not received before the connection wait timer expires.

Test Concept: Turn on the IUT. When the IUT attempts to connect to the TD as the primary hub or as a direct connection peer, the TD will accept the WebSocket connection but will not send a response to the connect request. It is verified that the IUT closes the WebSocket when the connection wait timer expires.

Configuration Requirements: The IUT is configured with the TD as the primary hub, or as a direct connect peer. The TD is configured to accept WebSocket connections but to not respond to Connect-Requests.

Test Steps:

1. MAKE(the IUT connect to the TD)
2. CHECK(that the IUT attempts to open a new WebSocket with the TD)
3. RECEIVE Connect-Request,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)

4. WAIT connection wait timeout
5. CHECK(that the IUT closed the WebSocket)

14.YY.1.2.6 HTTP 1.1 Fallback Test

Reference: YY.7

Purpose: To verify that the IUT is able to fallback to HTTP 1.1.

Test Concept: With the IUT configured to connect to the TD as the primary hub, turn on the IUT and when it attempts to connect, have the TD indicate that it only supports HTTP 1.1 as specified in RFC 6455. Verify that the IUT falls back to using HTTP 1.1.

Configuration Requirements: The IUT is configured to use HTTP 2 or later. If the IUT cannot be configured to do so, this test shall be skipped. The TD is configured to only support HTTP 1.1.

Test Steps:

1. MAKE(the IUT connect to the TD)
2. CHECK(that the IUT attempts to connect with HTTP 2 or later)
3. CHECK(that the IUT successfully connects to the TD using HTTP 1.1)
4. RECEIVE Connect-Request,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)
5. TRANSMIT Connect-Accept,
   'Message ID' = M1,
   -- '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)

14.YY.1.2.7 Rejection of Invalid Certificate Outgoing Connection Test

Reference: YY.7.4

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

Test Concept: With the IUT configured to connect to the TD as the primary hub or as a peer via a direct connection, allow the IUT to attempt the connection. The TD presents an invalid certificate during the connection. Verify that the WebSocket is not established.

Configuration Requirements: The TD is configured with an invalid certificate, or a certificate signed by a Certificate Authority which is not one recognized by the IUT. The TD shall be configured to accept the IUT's certificate.

Test Steps:

1. MAKE(the IUT connect to the TD)
2. CHECK(that the IUT initiated a WebSocket connection)
3. CHECK(that the WebSocket connection was failed by the IUT)

14.YY.1.2.8 No Additional Certificate Checks Performed Test On Outgoing Connections

Reference: YY.7.4

Purpose: Verify that the IUT does not apply checks beyond the 4 listed in YY.7.4 when not configured to do so.

Test Concept: With the IUT configured to connect to the TD as the primary hub, or as a peer via a direct connection, allow the IUT to attempt to connect. During the connection process the TD presents a valid certificate with field that would be caught if the IUT applied validation steps outside of those listed in YY.7.4.

Configuration Requirements: The IUT is configured to not perform any additional certificate checks beyond those identified in YY.7.4. The TD is configured with a certificate with a Common Name, Distinguished Name or Subject Alternate Name which does not match the TD device.

Test Steps:

1. MAKE(the IUT connect to the TD)
2. CHECK(that the IUT attempts to connect a WebSocket)
4. RECEIVE Connect-Request,
    'Message ID' = (M1: any valid value),
    -- 'Originating Virtual Address' absent
    -- 'Destination Virtual Address' absent
    'Destination Options' (absent or any valid value),
    -- '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)
5. TRANSMIT Connect-Accept,
    'Message ID' = M1,
    -- '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)

14.YY.1.2.9 Invalid WebSocket Data Test

Reference: YY.7.5.3

Purpose: To verify that the IUT will drop a WebSocket if a non-BVLC data frame is received on the WebSocket.

Test Concept: The IUT is connected to the network. Send a non-binary WebSocket data frame (the only type allowed for BVLC packets) to the IUT to hub WebSocket. Verify that the IUT closes the connection.

Configuration Requirements: The IUT is connected to the TD as a primary hub or direct connection peer.

Test Steps:

1. TRANSMIT WebSocket Frame,
    'opcode' = (a value in the range 1, 3 - 7), -- WebSocket opcode for a
    -- non-binary data frame
    Encapsulated-NPDU,
        'Message ID' = (M1: any valid value),
        -- 'Originating Virtual Address' absent
        -- 'Destination Virtual Address' absent
        -- 'Destination Options' absent
        'Data Options' = (X'01') -- Secure Path
        'BACnet NPDU' =
            ReadProperty-Request
            'Object Identifier' = (O: any object in the IUT),
            'Property Identifier' = Object_Name
2. CHECK(that the IUT closes the WebSocket)

14.YY.1.3 Basic Node Configuration Tests

14.YY.1.3.1 Configuration Via PEM Test

Reference: YY.7.4.1.3

Purpose: To verify that the IUT's configuration tool supports PEM format certificates.

Test Concept: The IUT's configuration tool is made to export a certificate signing request in PEM format. The PEM signing request is imported into the TD and a PEM format certificate is exported in PEM format. The PEM certificate is then loaded into the IUT with the IUT's configuration tool. The IUT is then configured to connect to the TD as the primary hub. The IUT is allowed to connect to the primary hub, and a successful connection is verified.

Test Steps:

1. MAKE(the IUT's configuration tool export a certificate signing request in PEM format)
2. CHECK(that the PEM file is well formed)
3. MAKE(import the PEM file into the TD and generate a PEM format certificate)
4. MAKE(the IUT's configuration tool load the PEM format certificate into the IUT)
5. MAKE(the IUT connect to the TD using the new certificate)

14.YY.1.3.2 Configuration Tool Accepts Arbitrary Valid Certificate Parameters Test

Reference: YY.7.4.1.3

Purpose: To verify that the IUT's configuration tool accepts arbitrary valid certificate parameters.

Test Concept: The tester selects arbitrary valid values for the certificate inputs and it is verified that the configuration tool accepts them.

Test Steps:

1. MAKE(the IUT's configuration tool export a certificate signing request using the tester selected certificate parameters in PEM format)
2. CHECK(that the PEM file is well formed)
3. MAKE(import the PEM file into the TD and generate a PEM format certificate)
4. MAKE(the IUT's configuration tool load the PEM format certificate into the IUT)
5. MAKE(the IUT connect to the TD using the new certificate)

14.YY.1.3.3 Factory Defaults Test

Reference: YY.7.4.2

Purpose: To verify that the IUT can be reset to factory defaults and that operational certificates and sensitive data are removed.

Test Concept: The IUT is configured to connect to the BACnet/SC network. Verify that the IUT can connect. Using the IUT's documented method for returning it to factory defaults, reset the IUT to factory defaults. Reset the IUT. Using the IUT's configuration tool or another vendor identified method, verify that the IUT no longer has its private data. Reset the IUT and verify that it does not attempt to connect to the previously configured hub. If the IUT accepts direct connections, attempt to connect to the previously configured IUT direct connect URL. Verify that the connection does not succeed. If the IUT was configured as a hub, verify that it does not accept connections using a previously acceptable certificate.

Test Steps:

1. MAKE(the IUT connect to the TD)
2. READ ON = (Device, IUT), Object_Name
3. MAKE(the IUT reset to factory defaults)
4. CHECK(that any private data viewable through the configuration tool has been discarded, including the hub URIs)
5. MAKE(reset the IUT)
6. MAKE(set the primary hub URI to the TD's hub URI)
7. MAKE(if the BACnet/SC Network Port has been disabled, enable it)
8. IF (the BACnet/SC port works when enabled without providing it with a certificate) THEN {
   CHECK(that the IUT is not able to connect to the TD due to not having a certificate signed by a mutually agreed upon CA)
   }

14.YY.2 Hub Tests

This clause contains test for BACnet/SC hub functions, both primary and failover.

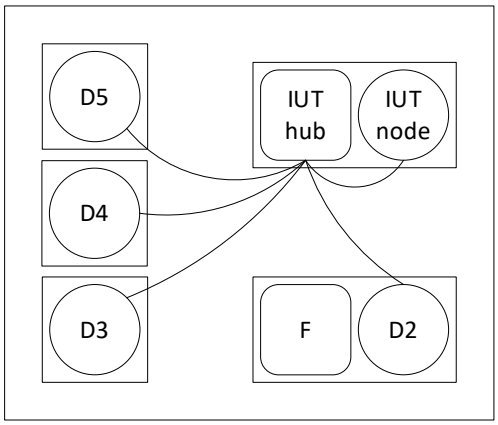


Figure X3: Network setup for hub function tests when the IUT is playing the role of the primary hub.

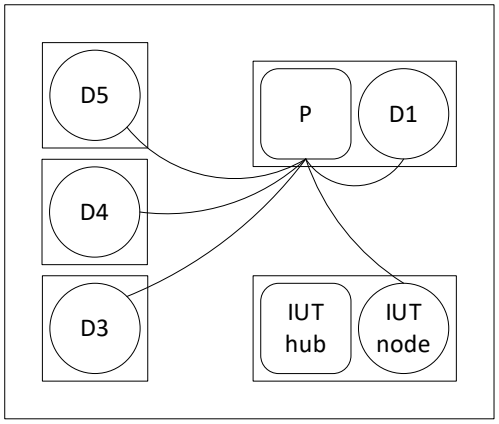


Figure X4: Network setup for hub function tests when the IUT is playing the role of the failover hub.

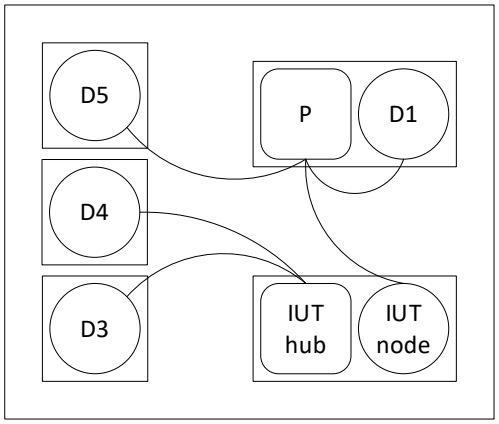


Figure X5: Network setup for hub function tests when the IUT is playing the role of the failover hub and not all nodes have reconnected to the primary.

14.YY.2.1 Hub Positive Tests

14.YY.2.1.1 Local Broadcast Initiation Test

Purpose: To verify that the IUT, as a hub, correctly initiates broadcast messages.

Test Concept: With IUT operating as a hub, make the IUT's node generate a broadcast message. Verify that the broadcast is sent correctly to all nodes attached to the hub.

Configuration Requirements: Configure the IUT as a hub. If the IUT does not support initiation of broadcast messages, this test shall be skipped.

1. MAKE(the IUT send a broadcast NPDU)
2. REPEAT Dx = (D2, D3, D4) {
   RECEIVE PORT (Dx-IUT hub WebSocket),
   Encapsulated-NPDU,
   'Originating Virtual Address' = (IUT's VMAC),
   'Destination Virtual Address' = X'FFFFFFFF' -- the local broadcast VMAC
   'Destination Options' = (absent or any valid value),
   'Data Options' = (secure path, plus 0 more data options),
   'Payload' = (a well formed NPDU)
   }
3. CHECK(that all of the NPDUs received in the preivous step are the same)

14.YY.2.1.2 Local Broadcast Execution Test

Reference: YY.5.3.3

Purpose: To verify that IUT, as a hub, correctly accepts and processes broadcast messages.

Test Concept: With the IUT operating as a hub, send a broadcast to the hub. Verify that the message is forwarded to all hub connectors except the one that originated it. Also verify that the hub's local node processes the broadcast.

Configuration Requirements: The IUT is operating as a hub and devices D2, D3 and D4 are connected to it.

Test Steps:

1. TRANSMIT PORT (D4-IUT hub WebSocket),
   Encapsulated-NPDU,
   -- 'Originating Virtual Address' absent
   'Destination Virtual Address' = X'FFFFFFFF' -- the local broadcast VMAC
   -- 'Destination Options' absent
   'Data Options' = (X'01'), -- secure path
   'Payload'
   Who-Is-Request
2. REPEAT Dx = (D2, D3) {
   RECEIVE PORT (Dx-IUT hub WebSocket),
   Encapsulated-NPDU,
   'Originating Virtual Address' = (D4's VMAC),
   'Destination Virtual Address' = X'FFFFFFFF' -- the local broadcast VMAC
   -- 'Destination Options' absent
   'Data Options' = (X'01'), -- secure path
   'Payload'
   Who-Is-Request
   }
3. RECEIVE PORT (D4-IUT hub WebSocket),
   Encapsulated-NPDU,
   'Originating Virtual Address' = (IUT's VMAC)
   'Destination Virtual Address' = (D4's VMAC or X'FFFFFFFF', the local broadcast VMAC)
   -- 'Destination Options' absent
   'Data Options' = (X'01'), -- secure path
   'Payload'
   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)

Notes to Tester: The order of the broadcasts sent by the hub, and the I-Am response can be sent in any order.

14.YY.2.1.3 Minimum NPDU Forwarding Size Test

Reference: YY.5.1

Purpose: To verify that the hub can forward BVLC messages of length 1497 with 4192 octets of data and destination options.

Test Concept: With the IUT operating as the primary hub, connect devices D3 and D4 to the IUT. D3 sends a BVLC of length 1497 octets with 4192 octets of data options to D4 via the hub. Verify that the BVLC is correctly forwarded to D4.

Configuration Requirements: The IUT is configured as a primary or failover hub and the test devices D3, D4 and D5 are connected to it.

Test Steps:

1. TRANSMIT PORT (D3-IUT hub WebSocket),
   Encapsulated-NPDU,
   -- 'Originating Virtual Address' absent
   'Destination Virtual Address' = (D4's VMAC),
   -- 'Destination Options' absent
   'Data Options' = (X'81',
   X'3F105A000034…'
   ), -- replace … with 4186 octets of any value
   'Payload'
   WriteProperty-Request,
   'Object Identifier' = (O: any object identifier),
   'Property Identifier' = (P: any property identifier),
   'Property Value' = (V: any data value with an encoded length which makes the PayLoad 1497 octets)
2. RECEIVE PORT (D4-IUT hub WebSocket),
   Encapsulated-NPDU,
   'Originating Virtual Address' = (D3's VMAC),
   -- 'Destination Virtual Address' absent
   -- 'Destination Options' absent
   'Data Options' = (X'81',
   X'3F105A000034…'
   ), -- replace … with 4186 octets of any value
   'Payload'
   WriteProperty-Request,
   'Object-Identifier' = O,
   'Property-Identifier' = P,
   'Property Value' = V

14.YY.2.1.4 Failover Hub Connects to Primary Hub Test

Reference: YY.5.2

Purpose: To verify that when the IUT is operating as a failover hub, the IUT's node connects to the primary hub.

Test Concept: The IUT is configured as a failover hub and the IUT's node has another hub set as its primary hub. Verify that the IUT connects to its primary hub.

Configuration Requirements: The IUT is configured as a failover hub with the TD set as its primary hub.

Test Steps:

1. MAKE(the IUT connect to the primary hub)
2. CHECK(that the IUT opens a WebSocket with the TD)
3. RECEIVE PORT (IUT-TD hub WebSocket),
   Connect-Request,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)

4. TRANSMIT PORT (IUT-TD hub WebSocket),
   Connect-Accept,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)

14.YY.2.1.5 Failover Hub's Local Node Connects to Failover Hub Test

Reference: YY.5.2

Purpose: To verify that, when operating as a failover hub, the IUT's local node connects to the local hub connection when the primary is offline.

Test Concept: The IUT is configured as the IUT's local node's failover hub and the IUT's local node has the TD set as its primary hub. Verify that the IUT connects to its primary hub. Disconnect the primary from the network. Wait for the IUT to recognize the primary is offline. Connect D3 to the IUT's hub connection. Make the IUT's local node send a message to D3. Verify that the message is sent from the IUT's hub connection.

Configuration Requirements: The IUT is configured as a failover hub with the TD set as its local node's primary hub. The TD is configured to be a hub.

1. MAKE(the IUT connect to the primary hub)
2. CHECK(that the IUT opens a WebSocket with the TD)
3. RECEIVE PORT (IUT-TD hub WebSocket),
   Connect-Request,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)
4. TRANSMIT PORT (IUT-TD hub WebSocket),
   Connect-Accept,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' (absent or any valid value),
   -- '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)
5. WAIT a tester selected amount of time
6. MAKE(the TD's hub function stop)
7. WAIT 2 times the IUT's HeartBeat timeout
8. MAKE(D3 open a WebSocket with IUT's hub URI)
9. TRANSMIT PORT (D3-IUT hub 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 D1's maximum BVLC accepted length),
'Maximum NPDU Length' = (the D1's maximum NPDU accepted length)

10. RECEIVE PORT (D3-IUT hub WebSocket),
Connect-Accept,
'Message ID' = M2,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' (absent or any valid value),
-- '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)
11. MAKE(the IUT's node send an NPDU to D1)
12. RECEIVE PORT (D3-IUT hub WebSocket),
Encapsulated-NPDU,
'Message ID' = (M3: any valid value),
'Originating Virtual Address' = (IUT's VMAC),
'Destination Virtual Address' = (absent or local broadcast)
'Destination Options' (absent or any valid value),
'Data Options' = (secure path + 0 or more valid data options),
'Payload' = (any valid NPDU)

14.YY.2.1.6 Failover Hub Split Horizon Test

Reference: YY.5.2

Purpose: To verify that the IUT, operating as a failover hub, continues to perform hub functions when the IUT node is connected to the primary hub.

Test Concept: The IUT is configured as a failover hub and is connected to the TD acting as the primary hub. Connect device D3 to the IUT's hub URI. Verify that the IUT accepts the connection. Connect device D4 to the IUT's hub function. Verify that the IUT accepts the connection. D3 sends a broadcast to the IUT's hub function. Verify that the broadcast is properly forwarded to D4 and no other nodes. Verify that the IUT's node does not receive the broadcast. Connect D5 to the primary hub. D5 sends an Advertisement-Solicitation message to ensure that the IUT is aware of its existence. D3 sends a unicast destined for D5 to the hub for forwarding. Verify that the IUT does not forward the request.

Configuration Requirements: The IUT is configured as a failover hub with the TD set as its primary hub. The TD is configured to be a primary hub. The IUT's node is connected to the primary hub.

1. CHECK(that the IUT's node is connected to the TD's hub function)
2. MAKE(D3 open a WebSocket with IUT's hub URI)
3. TRANSMIT PORT (D3-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' = (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)
4. RECEIVE PORT (D3-IUT hub WebSocket),
Connect-Accept,
'Message ID' = M1,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' (absent or any valid value),

```
        -- '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)
5.  TRANSMIT PORT (D4-IUT hub 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' =            (D4's VMAC),
        'Device UUID' =             (D4's UUID),
        'Maximum BVLC Length' =     (the D4's maximum BVLC accepted length),
        'Maximum NPDU Length' =     (the D4's maximum NPDU accepted length)
6.  RECEIVE PORT (D4-IUT hub WebSocket),
        Connect-Accept,
        'Message ID' =              M2,
        -- 'Originating Virtual Address' absent
        -- 'Destination Virtual Address' absent
        'Destination Options'       (absent or any valid value),
        -- '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)

-- verify that D3 and D4 can communicate with each other and that the IUT's local node does not receive
--  broadcasts generated by D3
7.  TRANSMIT PORT (D3-IUT hub WebSocket),
        Encapsulated-NPDU,
        'Message ID' =              (M3: any valid value),
        -- 'Originating Virtual Address' absent
        'Destination Virtual Address' =  (X'FFFFFFFF' local broadcast VMAC),
        -- 'Destination Options' absent
        'Data Options' =            (X'01'),      -- secure path
        'Payload' =
            DNET =                  GLOBAL_BROADCAST,
            Who-Is-Request
8.  RECEIVE PORT (D4-IUT hub WebSocket),
        Encapsulated-NPDU,
        'Message ID' =              (M3: any valid value),
        'Originating Virtual Address' =  (D3's VMAC),
        'Destination Virtual Address' =  (X'FFFFFFFF' local broadcast VMAC),
        -- 'Destination Options' absent
        'Data Options' =            (X'01'),      -- secure path
        'Payload' =
            DNET =                  GLOBAL_BROADCAST,
            Who-Is-Request
9.  CHECK(that the IUT does not generate an I-Am for its local node)

-- verify that D5, connected to the primary, and the IUT's node can communicate
10. TRANSMIT PORT (IUT-TD hub WebSocket),
        Advertisement-Solicitation,
        'Message ID' =              (M4: any valid value),
        'Originating Virtual Address' =  (D5's VMAC),
        -- 'Destination Virtual Address' absent
        -- 'Destination Options' absent
        -- 'Data Options' absent
```

11. RECEIVE PORT (IUT-TD hub WebSocket),
    Advertisement,
    'Message ID' = (M5: any valid value),
    -- 'Originating Virtual Address' absent
    'Destination Virtual Address' = (D5's VMAC),
    'Destination Options' (absent, or a valid list of options),
    -- 'Data Options' absent
    'Payload'
        'Hub Connection Status' = X'01', -- connected to the primary hub
        'Accept Direct Connections' = (as per the IUT's configuration),
        'Maximum BVLC Length' = (as per the IUT's configuration),
        'Maximum NPDU Length' = (as per the IUT's configuration)

-- verify that D3, connected to the failover, and D5, connected to the primary, cannot communicate
12. TRANSMIT PORT (D3-IUT hub WebSocket),
    Advertisement-Solicitation,
    'Message ID' = (M6: any valid value),
    -- 'Originating Virtual Address' absent
    'Destination Virtual Address' = (D5's VMAC),
    -- 'Destination Options' absent
    -- 'Data Options' absent
13. CHECK(that the IUT does not forward the Advertisement-Solicitation)

-- verify that D3, connected to the failover, and the IUT's node, connected to the primary, cannot communicate
14. TRANSMIT PORT(D3-IUT hub WebSocket),
    Advertisement-Solicitation,
    'Message ID' = (M7: any valid value),
    -- 'Originating Virtual Address' absent
    -- 'Destination Virtual Address' absent
    -- 'Destination Options' absent
    -- 'Data Options' absent
15. CHECK(that the IUT does not generate an Advertisement)

14.YY.2.1.7 Hub Forwards Unicast BVLCs Test

Reference: YY.5.3.2

Purpose: To verify that a hub correctly forwards unicast BVLCs received on any current hub connection.

Test Concept: The IUT is operating as the primary hub. D3 sends a unicast to D4 via the hub. Verify that the IUT correctly forwards the message to D4.

Configuration Requirements: The IUT is configured as the primary hub. D3 and D4 are connected to the IUT's hub function.

-- verify that D3 and D4 can communicate with each other
1. TRANSMIT PORT (D3-IUT hub WebSocket),
    Encapsulated-NPDU,
    'Message ID' = (M1: any valid value),
    -- 'Originating Virtual Address' absent
    'Destination Virtual Address' = (D4's VMAC),
    -- 'Destination Options' absent
    'Data Options' = (X'01'), -- secure path
    'Payload' =
        Who-Is-Request
2. RECEIVE PORT (D4-IUT hub WebSocket),
    Encapsulated-NPDU,
    'Message ID' = (M1: any valid value),
    'Originating Virtual Address' = (D3's VMAC),
    -- 'Destination Virtual Address' absent
    -- 'Destination Options' absent
    'Data Options' = (X'01'), -- secure path
    'Payload' =

Who-Is-Request
3. CHECK(that the Encapsulated-NPDU is not fowarded to any other devices, is not routed, and that the IUT does not generate an I-Am)

14.YY.2.1.8 No Additional Certificate Checks Performed Test On Incoming Connections

Reference: YY.7.4

Purpose: Verify that the IUT does not apply checks beyond the 4 listed in YY.7.4 when not configured to do so.

Test Concept: With the IUT configured to operate as a hub. The TD attempts to connect to the IUT with a valid certificate with field that would be caught if the IUT applied validation steps outside of those listed in YY.7.4.

Configuration Requirements: The IUT is configured to operate as a hub. The TD is configured with a certificate with a Common Name, Distinguished Name or Subject Alternate Name which does not match the TD device.

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)

14.YY.2.1.9 Duplicate Connection Test

Reference: YY.6.2, YY.6.2.1, YY.6.2.3

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. When the connection is complete, D3 attempts to bring up a second connection to the IUTs hub URI. Verify that the IUT accepts the second connect request and closes the first connection. Repeat the reconnection, but with a new VMAC for D3 and ensure that the new request is accepted and the existing one dropped.

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(D1 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)

14.YY.2.1.10 Heartbeat-Request Execution Test

Reference: YY.2.14, YY.2.15, YY.6.3

Purpose: To verify that the IUT accepts and responds to Heartbeat-Requests.

Test Concept: With the IUT operating as a hub, the TD connects to the IUT. The TD sends a Heartbeat-Request to the IUT. Verify that the IUT responds with a Heartbeat-ACK.

Configuration Requirements: The IUT is configured as a hub, and the TD is connected to it.

Test Steps:

1. TRANSMIT PORT (TD-IUT hub WebSocket),
   Heartbeat-Request,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   -- 'Destination Options' absent
   -- 'Data Options' absent
2. RECEIVE PORT (TD-IUT hub WebSocket),
   Heartbeat-ACK,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' = (absent or a valid list of options),
   -- 'Data Options' absent

14.YY.2.2 Hub Negative Tests

14.YY.2.2.1 Hub Discards BVLCs with Non-connected VMAC Test

Reference YY.5.1, YY.5.3.3

Purpose: To verify that a hub will drop received BVLCs with a destination VMAC not connected to the hub.

Test Concept: With the IUT operating as the primary hub, connect to the IUT and send a message to a VMAC which is not connected to the IUT. Verify that the BVLC is silently dropped.

Configuration Requirements: The IUT is configured as a hub and device D3 is connected but device D4 is not.

Test Steps:

1. TRANSMIT PORT (D3-IUT hub WebSocket),
   Advertisement-Solicitation,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   'Destination Virtual Address' = (D4's VMAC)
   -- 'Destination Options' absent
   -- 'Data Options' absent
2. CHECK(that the Advertisement-Solicitation is silently dropped)

14.YY.2.2.2 Connect-Request Wait Time Test

Reference: YY.6.2, YY.7.5.5

Purpose: To verify that the hub will close the WebSocket if the Connect-Request is not received before the connection wait timer expires.

Test Concept: With the IUT connected to the BACnet/SC network. Open a WebSocket connection with the IUT's hub port, but do not send a connect-request. Verify that the IUT closes the WebSocket after the connection wait timer expires.

Configuration Requirements: The IUT is configured to be a BACnet/SC hub.

Test Steps:

1. MAKE(a WebSocket connection to the IUT's hub function)
2. WAIT the connection wait timer expiration time
3. CHECK(that the IUT closed the WebSocket and did not send any messages on the WebSocket)

14.YY.2.2.3 VMAC Collision Detection Test

Reference: YY.6.2.1

Purpose: To verify that connections to devices with a duplicate VMAC are rejected.

Test Concept: With the IUT operating as the primary hub, connect device D3 to the IUT's hub function. Connect D4 to the IUT's hub function and provide the IUT's VMAC in the Connect-Request. Verify that the IUT NAKs the connect request with an error code of NODE_DUPLICATE_VMAC. Retry the Connect-Request with D3's VMAC. Verify that the IUT NAK's the connect request.

Configuration Requirements: The IUT is configured as a hub. D3 is connected to the IUT and D4 is not.

1. MAKE(D4 connect a WebSocket to the IUT's hub function)
2. TRANSMIT PORT (D4-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' = (IUT's VMAC),
   'Device UUID' = (D4's UUID),
   'Maximum BVLC Length' = (the D4's maximum BVLC accepted length),
   'Maximum NPDU Length' = (the D4's maximum NPDU accepted length)
3. RECEIVE PORT (D4-IUT hub WebSocket),
   BVLC-Result,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' = (absent or a list of valid options),
   -- 'Data Options' absent
   'Result for BVLC Function' = X'06', -- Connect-Request
   'Result Code' = X'01', -- NAK
   'Error Header Marker' = X'00', -- not a header option problem
   'Error Class' = COMMUNICATION,
   'Error Code' = NODE_DUPLICATE_VMAC
4. TRANSMIT PORT (D4-IUT hub 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' = (D4's UUID),
'Maximum BVLC Length' = (the D4's maximum BVLC accepted length),
'Maximum NPDU Length' = (the D4's maximum NPDU accepted length)

5. RECEIVE PORT (D4-IUT hub WebSocket),
BVLC-Result,
'Message ID' = M2,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' = (absent or a list of valid options),
-- 'Data Options' absent
'Result for BVLC Function' = X'06', -- Connect-Request
'Result Code' = X'01', -- NAK
'Error Header Marker' = X'00', -- not a header option problem
'Error Class' = COMMUNICATION,
'Error Code' = NODE_DUPLICATE_VMAC
6. TRANSMIT PORT (D4-IUT hub WebSocket),
Connect-Request,
'Message ID' = (M3: any valid value),
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
-- 'Destination Options' absent
-- 'Data Options' absent
'VMAC Address' = (D4's VMAC),
'Device UUID' = (D4's UUID),
'Maximum BVLC Length' = (the D4's maximum BVLC accepted length),
'Maximum NPDU Length' = (the D4's maximum NPDU accepted length)
7. RECEIVE PORT (D4-IUT hub WebSocket),
Connect-Accept,
'Message ID' = M3,
-- '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)

14.YY.2.2.4 Rejection of Invalid Certificate Incoming Connection Test

Reference: YY.7.4

Purpose: To verify that the IUT will not accept incoming connections if the peer's certificate is invalid.

Test Concept: With the IUT operating as a hub, D3 attempts to open a WebSocket to the IUT's hub URI. D3 presents an invalid certificate during the connection process. Verify that the WebSocket is not established.

Configuration Requirements: The IUT is configured to be a hub. The TD is configured with an invalid certificate, or a certificate signed by a Certificate Authority which is not one recognized by the IUT. The TD shall be configured to accept the IUT's certificate.

Test Steps:

1. MAKE(D3 attempt to connect to the IUT's hub function with an invalid certificate)
2. CHECK(that the IUT refused the connection)

14.YY.3 Direct Connect Tests

This group of tests verifies secure connect devices using direct connections. The logical configuration of the network used for these tests is shown in Figure X6. The test descriptions in this clause assume that the TD plays the role of all of the other devices in the network configuration. For the tests in this section, unless specified through a PORT parameter, messages specified by the test are on the IUT to peer device direct connection WebSocket.

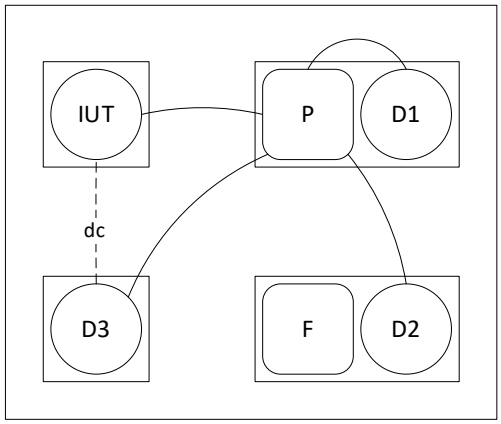


Figure X6: Network setup for basic node tests showing connections when the primary hub is active.

14.YY.3.1 Direction Connect Basic Tests

14.YY.3.1.1 Direction Connect Basic Positive Tests

14.YY.3.1.1.1 Unicast Through Direct Connect Test

Purpose: Verify that the IUT correctly composes unicasts aimed at a device through a direct connect.

Test Concept: With the IUT connected to D3 via a direct connection, have D3 send a request to the IUT which requires a response and verify that the response is sent back through the direct connection. If the IUT initiates APDU requests, make the IUT initiate a request and verify that it is transmitted through the direct connection.

Configuration Requirements: The IUT is connected to D3 via direct connect.

Test Steps:

1. TRANSMIT PORT (IUT-D3 direct connect WebSocket),
   Encapsulated-NPDU,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   -- 'Destination Options' absent
   'Data Options' = (X'01'), -- secure path
   'Payload'
   ReadProperty-Request,
   'Object Identifier' = (O: any object in the IUT),
   'Property Identifier' = (P: any readable property in O)
2. RECEIVE PORT (IUT-D3 direct connect WebSocket),
   Encapsulated-NPDU,
   'Message ID' = (M2: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   -- 'Destination Options' absent
   'Data Options' = (X'01'), -- secure path
   'Payload'
   ReadProperty-ACK,
   'Object Identifier' = (O: any object in the IUT),
   'Property Identifier' = (P: any readable property in O),
   'Property Value' = (V: any valid value)

14.YY.3.1.1.2 Direct Connect Disconnect Test

Reference: YY.2.12, YY.2.7

Purpose: To verify that the IUT is able to correctly disconnect a direct connection with a non-hub peer BACnet/SC node.

Test Concept: With the IUT connected to another device via a direct connection, make the IUT close the connection. Verify that the IUT correctly requests and performs a disconnect.

Configuration Requirements: The IUT is directly connected to D3. If the IUT never closes established direct connections, this test shall be skipped.

Test Steps:

1. MAKE(the IUT drop the direct connection to D3)
2. RECEIVE PORT (IUT-D3 direct connect WebSocket)
   Disconnect-Request,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   'Destination Options' = (absent or a list of valid options),
   -- 'Data Options' absent
3. TRANSMIT PORT (IUT-D3 direct connect WebSocket)
   Disconnect-ACK,
   'Message ID' = M1,
   --'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   -- 'Destination Options' absent
   -- 'Data Options' absent
4. CHECK(that the WebSocket is closed)

14.YY.3.1.1.3 Direct Connect Establishment Failover Test

Reference: YY.1.6

Purpose: To verify that the IUT is able to continue to communicate with an arbitrary peer BACnet/SC node via the hub after a direct connect request to the peer is rejected.

Test Concept: With IUT connected to the network, and with a direct connection between the IUT and D3, make the IUT communicate with D3 to verify that the connection is working. Make D3 drop the direct connection. Make the IUT communicate with D3, having the IUT initiate the conversation if possible. Verify that the IUT is able to communicate with D3 through the hub.

Configuration Requirements: The IUT is connected to the primary hub and has a direct connection to D3. D3 shall be configured such that it will reject any attempts to re-establish the direct connection after the initial direct connection is dropped.

Test Steps:

1. TRANSMIT PORT (IUT-D3 direct connect WebSocket),
   Advertisement,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   -- 'Destination Options' absent
   -- 'Data Options' absent
   'Payload'
      'Hub Connection Status' = X'01', -- connected to the primary
      'Accept Direct Connections' = X'00', -- does not accept incoming direct connections.
      'Maximum BVLC Length' = (any valid value),
      'Maximum NPDU Length' = (any valid value)
2. TRANSMIT PORT (IUT-D3 direct connect WebSocket),
   Disconnect-Request,
   'Message ID' = (M2: any valid value),
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' absent
   -- 'Destination Options' absent

```
        -- 'Data Options' absent
3.  RECEIVE PORT IUT to D3 direct connect WebSocket),
        Disconnect-ACK,
        'Message ID' =                  M2,
        -- 'Originating Virtual Address' absent
        -- 'Destination Virtual Address' absent
        -- 'Destination Options' absent
        -- 'Data Options' absent
4.  IF the IUT can initiate an NPDU request to D3 THEN {
        MAKE(IUT send an NPDU to D3)
        -- note that the IUT might attempt a reconnect at this point, which will be rejected by D3, and that such an
        --  attempt is acceptable
        RECEIVE PORT (IUT-TD hub WebSocket),
            Encapsulated-NPDU,
            'Message ID' =                  (M3: any valid value),
            -- 'Originating Virtual Address' absent
            'Destination Virtual Address' = D3,
            'Destination Options' =         (absent or a valid list of options),
            'Data Options' =                (secure path plus 0 or more valid data options),
            'Payload' =                     (a valid NPDU)
    } ELSE {
        TRANSMIT PORT (IUT-TD hub WebSocket),
            Encapsulated-NPDU,
            'Message ID' =                  (M4: any valid value),
            'Originating Virtual Address' = D3,
            -- 'Destination Virtual Address' absent
            -- 'Destination Options' absent
            'Data Options' =                (X'01') ,       -- secure path
            'Payload' =
                ReadProperty-Request,
                'Object Identifier' =       (O: any object in the IUT),
                'Property Identifier' =     (P: any property in O)
        RECEIVE PORT (IUT-TD hub WebSocket),
            Encapsulated-NPDU,
            'Message ID' =                  (M5: any valid value),
            -- 'Originating Virtual Address' absent
            'Destination Virtual Address' = D3,
            'Destination Options' =         (absent or a valid list of options),
            'Data Options' =                (secure path plus 0 or more valid data options),
            'Payload' =
                ReadProperty-ACK,
                'Object Identifier' =       (O: any object in the IUT),
                'Property Identifier' =     (P: any property in O),
                'Property Value' =          (any valid value)
```

14.YY.3.1.2 Direction Connect Basic Negative Tests

14.YY.3.1.2.1 Discard Broadcast BVLC Received on Direct Connect Test

Reference: YY.4.2.2

Purpose: To verify that broadcast BVLCs received on a direct connection are discarded.

Test Concept: With the IUT connected to another device, D3, via a direct connection, have D3 send a broadcast to the IUT. Verify that the IUT does not process the broadcast.

Configuration Requirements: The IUT is connected to the SC network and has a direct connection to D3.

Test Steps:

```
1.  TRANSMIT PORT (IUT-D3 direct connect WebSocket)

        Encapsulated-NPDU,
```

'Message ID' = (M1: any valid value),
-- 'Originating Virtual Address' absent
'Destination Virtual Address' = X'FFFFFFFF', -- the local broadcast VMAC
'Destination Options' = (absent or a valid list of options),
'Data Options' = (secure path plus 0 or more valid data options),
'Payload' =
Who-Is-Request

2. CHECK(that the IUT does not generate a BVLC-Response nor an I-Am-Request)

14.YY.3.2 Accepting Direct Connect Tests

14.YY.3.2.1 Accepting Direct Connect Positive Tests

14.YY.3.2.1.1 Direct Connect Acceptance Test

Reference: YY.2.6, YY.2.7

Purpose: To verify that the IUT correctly accepts direct connections.

Test Concept: With IUT connected to the network, have peer node D3 establish a direct connection with the IUT. Verify that the IUT correctly accepts the connection.

Configuration Requirements: The IUT is configured with a single WebSocket-URI at which it accepts direct connections. The IUT is connected to the primary hub.

1. TRANSMIT PORT (IUT-TD hub WebSocket)
Address-Resolution,
'Message ID' = (M1: any valid value),
'Originating Virtual Address' = D3,
-- 'Destination Virtual Address' absent
-- 'Destination Options' absent
-- 'Data Options' absent
2. RECEIVE PORT (IUT-TD hub WebSocket)
Address-Resolution-ACK,
'Message ID' = M1,
-- 'Originating Virtual Address' absent
'Destination Virtual Address' = D3,
'Destination Options' = (absent or a valid list of options),
-- 'Data Options' absent
'Payload'
'WebSocket-URIs' (W: the configured WebSocket URI for the IUT)
}
3. MAKE(D1 connected a WebSocket to the IUT)
4. TRANSMIT PORT (D3-IUT direct connect 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 D1's maximum BVLC accepted length),
'Maximum NPDU Length' = (the D1's maximum NPDU accepted length)
6. RECEIVE PORT (D3-IUT direct connect WebSocket),
Connect-Accept,
'Message ID' = M2,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' (absent or any valid value),
-- '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)

14.YY.3.2.1.2 No Additional Certificate Checks Performed Test On Incoming Connections

Reference: YY.7.4

Purpose: Verify that the IUT does not apply checks beyond the 4 listed in YY.7.4 when not configured to do so.

Test Concept: With the IUT configured to accept direct connections. The TD attempts to connect to the IUT with a valid certificate with field that would be caught if the IUT applied validation steps outside of those listed in YY.7.4.

Configuration Requirements: The IUT is configured to accept direct connections. The TD is configured with a certificate with a Common Name, Distinguished Name or Subject Alternate Name which does not match the TD device.

Test Steps:

1. MAKE(the TD open a WebSocket to the IUT's direct connect URI)
2. TRANSMIT PORT (TD-IUT direct connect 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 direct connect 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)

14.YY.3.2.2 Accepting Direct Connect Negative Tests

14.YY.3.2.2.1 Connect-Request Wait Time Test

Reference: YY.6.2, YY.7.5.5

Purpose: To verify that the IUT will close the WebSocket if the Connect-Request is not received before the connection wait timer expires.

Test Concept: With the IUT connected to the BACnet/SC network, open a WebSocket connection to the IUT's direct connect URI, but do not send a connect-request. Verify that the IUT closes the WebSocket after the connection wait timer expires.

Configuration Requirements: The IUT is configured to accept direct connections.

1. MAKE(a WebSocket connection to the IUT's direct connect WebSocket-URI)
2. WAIT the connection wait timer expiration time
3. CHECK(that the IUT closed the WebSocket and did not send any messages on the WebSocket)

14.YY.3.2.2.2 Direct-Connect Duplicate Connection for IUT Accepted Connections Test

Reference: YY.6.2.3

Purpose: To verify that duplicate direct connect connection requests result in the original connection being dropped for connections initiated by the IUT's peer.

Test Concept: With the IUT connected to the BACnet/SC network, D3 connects to the IUT's direct connect URI. When the connection is complete, D3 attempts to bring up a second connection to the IUTs direct connect URI. Verify that the IUT accepts the second connect request and closes the first connection. Repeat the reconnection, but with a new VMAC for D3 and ensure that the new request is accepted and the existing one dropped.

Configuration Requirements: The IUT is configured to accept direct connections.

Test Steps:

1. MAKE(D3 connect to the IUT's direct connect WebSocket URI)
2. TRANSMIT PORT (D3-IUT direct connect 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 direct connect 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 direct connect WebSocket-URI)
5. TRANSMIT PORT (D3-IUT direct connect 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 direct connect 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 direct connect 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 direct connect 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 direct connect WebSocket URI)
11. TRANSMIT PORT (D3-IUT direct connect third 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 direct connect 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 direct connect 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 direct connect 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 direct connect WebSocket)

14.YY.3.2.2.3 Direct-Connect Duplicate Connection for IUT Initiated Connections Test

Reference: YY.6.2.3

Purpose: To verify that duplicate direct connect connection requests result in the original connection being dropped when the original connection was initiated by the IUT.

Test Concept: With the IUT connected to the BACnet/SC network, D3 connects to the IUT's direct connect URI. When the connection is complete, D3 attempts to bring up a second connection to the IUTs direct connect URI. Verify that the IUT accepts the second connect request and closes the first connection. Repeat the reconnection, but with a new VMAC for D3 and ensure that the new request is accepted and the existing one dropped.

Configuration Requirements: The IUT is configured to initiate and accept direct connections. If the IUT does not support both initiating and accepting direct connections, this test shall be skipped. If the IUT does not support 2 or more simultaneous direct connect WebSocket connections, this test shall be skipped.

Test Steps:

1. MAKE(IUT connect to D3's WebSocket URI)
2. RECEIVE PORT (IUT-D3 direct connect 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' = (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)

3. TRANSMIT PORT (IUT-D3 direct connect first WebSocket),
   Connect-Accept,
   'Message ID' = M1,
   -- 'Originating Virtual Address' absent
   -- 'Destination Virtual Address' 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)
4. MAKE(D3 connect a second WebSocket to the IUT's direct connect WebSocket-URI)
5. TRANSMIT PORT (D3-IUT direct connect 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 direct connect 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 direct connect 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 direct connect 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 direct connect WebSocket URI)
11. TRANSMIT PORT (D3-IUT direct connect third 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 direct connect 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 direct connect 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 direct connect 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 direct connect WebSocket)

14.YY.3.2.2.4 VMAC Collision Detection Test

Reference: YY.6.2.1

Purpose: To verify that connections to devices with a duplicate VMAC are rejected.

Test Concept: With the IUT connected to the network, have D3 open a WebSocket to the IUT's direct connect URI and provide the IUT's VMAC in the connect request. Verify that the IUT NAKs the connect request with an error code of NODE_DUPLICATE_VMAC.

1. MAKE(D3 connect a WebSocket to the IUT's direct connect WebSocket URI)
2. TRANSMIT PORT (D3-IUT direct connect 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' = (IUT'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 direct connect WebSocket),
    BVLC-Result,
    'Message ID' = M1,
    -- 'Originating Virtual Address' absent
    -- 'Destination Virtual Address' absent
    'Destination Options' = (absent or a list of valid options),

-- 'Data Options' absent
'Result for BVLC Function' = X'06', -- Connect-Request
'Result Code' = X'01', -- NAK
'Error Header Marker' = X'00', -- not a header option problem
'Error Class' = COMMUNICATION,
'Error Code' = NODE_DUPLICATE_VMAC

4. TRANSMIT PORT (D3-IUT direct connect 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)
5. RECEIVE PORT (D3-IUT direct connect 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)

14.YY.3.2.2.5 Rejection of Invalid Certificate Incoming Connection Test

Reference: YY.7.4

Purpose: To verify that the IUT will not accept incoming connections if the peer's certificate is invalid.

Test Concept: With the IUT connected to the network, D3 attempts to open a WebSocket to the IUT's direct connect URI. D3 presents an invalid certificate during the connection process. Verify that the WebSocket is not established.

Configuration Requirements: The IUT is configured to accept direct connections. D3 is configured with an invalid certificate. D3 is configured to accept the IUT's certificate.

Test Steps:

1. MAKE(D3 attempt to connect to the IUT's direct connect URI with an invalid certificate)
2. CHECK(that the IUT refused the connection)

14.YY.3.3 Initiating Direct Connect Tests

14.YY.3.3.1 Initiating Direct Connect Positive Tests

14.YY.3.3.1.1 Direct Connect Establishment Test

Reference: YY.2.6, YY.2.7

Purpose: To verify that the IUT is able to correctly establish a direct connection with a non-hub peer BACnet/SC node.

Test Concept: With IUT connected to the network, make the IUT establish a direct connection to another node on the network. Verify that the direct connection is correctly established.

Configuration Requirements: The IUT is configured to support establishing direct connections. The IUT is connected to the primary hub. D3 is configured to support accepting direct connections. The IUT is configured to use dynamic discovery of WebSocket-URIs if supported, otherwise the WebSocket-URI for D3 is configured into the IUT.

Test Steps:

1. MAKE(the IUT establish a direct connection to D3)
2. IF (the IUT supports discovering direct connection URIs) {

RECEIVE PORT (IUT-TD hub WebSocket)
Address-Resolution,
'Message ID' = (M1: any valid value),
-- 'Originating Virtual Address' absent
'Destination Virtual Address' = D3,
'Destination Options' = (absent or a list of valid options),
-- 'Data Options' absent
TRANSMIT PORT (IUT-TD hub WebSocket)
Address-Resolution-ACK,
'Message ID' = M1,
'Originating Virtual Address' = D3
-- 'Destination Virtual Address' absent
-- 'Destination Options' absent
-- 'Data Options' absent
'Payload'
'WebSocket-URIs' (W: a WebSocket URI which D3 can be reached at)
}

4. CHECK(that the IUT opens a WebSocket to D3's WebSocket-URI)
5. RECEIVE PORT (IUT-D3 direct connect WebSocket),
Connect-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
'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)
6. TRANSMIT Connect-Accept,
'Message ID' = M2,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' (absent or any valid value),
-- '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)

14.YY.3.3.1.2 Direct Connect Multiple URI Test

Reference: YY.2.12, YY.2.7, YY.3.3

Purpose: To verify that the IUT is able to correctly connect to a peer with multiple URIs where only 1 of the URIs is valid for IUT.

Test Concept: With IUT connected to the network, make the IUT establish a direct connection to D3 with D3 configured to advertise multiple URIs of which only 1 is valid for the IUT. Verify that the direct connection is correctly established.

Configuration Requirements: The IUT is configured to support establishing direct connections. The IUT is connected to the primary hub. D3 is configured to support accepting direct connections and is configured with multiple WebSocket-URIs. Only one of the URIs configured into D3 shall be reachable by the IUT and that URI shall not be the first or last URI in the D3's list of URIs. The IUT is configured to use dynamic discovery of WebSocket-URIs. If the IUT does not support dynamic discovery of WebSocket-URIs, this test shall be skipped.

Test Steps:

1. MAKE(the IUT attempt a direct connection to D3)
2. RECEIVE PORT (IUT-TD hub WebSocket)
Address-Resolution,
'Message ID' = (M1: any valid value),
-- 'Originating Virtual Address' absent

'Destination Virtual Address' = D3,
'Destination Options' = (absent or a list of valid options),
-- 'Data Options' absent
TRANSMIT PORT (IUT-TD hub WebSocket)
Address-Resolution-ACK,
'Message ID' = M1,
'Originating Virtual Address' = D3
-- 'Destination Virtual Address' absent
-- 'Destination Options' absent
-- 'Data Options' absent
'Payload'
'WebSocket-URIs' (W1, …, Wi, …, Wn) -- only Wi is a WebSocket-URI that
-- will connect to D3
}

4. CHECK(that the IUT repeatedly attempts to connect to the WebSocket-URIs until Wi is used)
5. RECEIVE PORT (IUT-D3 direct connect WebSocket),
Connect-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
'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)
6. TRANSMIT PORT (IUT-D3 direct connect WebSocket),
Connect-Accept,
'Message ID' = M2,
-- 'Originating Virtual Address' absent
-- 'Destination Virtual Address' absent
'Destination Options' (absent or any valid value),
-- '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)

14.YY.3.3.2 Initiating Direct Connect Negatives Tests

14.YY.3.3.2.1 Invalid Web Socket Scheme In Configured Direct Connect URI Test

Reference: YY.7.5.1

Purpose: To verify that the IUT does not attempt to connect to configured direct connect URIs with invalid schemes.

Test Concept: The IUT is configured with the direct connect URI for device D3 and is connected to the BACnet/SC network. Take the steps that would normally cause the IUT initiate a direct connect to D3. Verify that the IUT does not attempt to connect to the invalid URI.

Configuration Requirements: The IUT is configured to support establishing direct connections to D3 and D3's URI is configured into the IUT so it does not have to dynamically discover it. The configured URI shall have an invalid websocket scheme. The IUT starts the test already connected to the primary hub. D3 is configured to support accepting direct connections. If the IUT does not support configured peer URIs or does not accept invalid schemes in configured URIs, this test shall be skipped.

Test Steps:

1. MAKE(the IUT initiate the establishement of a direct connection to D3)
2. CHECK(that the IUT does not attempt to open a WebSocket to the invalid WebSocket-URI)

14.YY.3.3.2.2 Invalid Web Socket Scheme in Discovered Direct Connect URI Test

Reference: YY.7.5.1

Purpose: To verify that the IUT does not attempt to connect to discovered URIs containing invalid websocket schemes.

Test Concept: With the IUT connected to the BACnet/SC network, make the IUT initiate a direct connect to D3. D3 advertises its direct connect URL with an invalid scheme. Verify that the IUT does not attempt to connect to the invalid URI.

Configuration Requirements: The IUT is configured to support establishing direct connections and is connected to the primary hub. D3 is configured to support accepting direct connections. If the IUT does not support dynamically determining direct connect URIs, this test shall be skipped.

Test Steps:

1. MAKE(the IUT initiate the establishement of a direct connection to D3)
2. IF (the IUT supports discovering direct connection URIs) {
   RECEIVE PORT (IUT-TD hub WebSocket)
   Address-Resolution,
   'Message ID' = (M1: any valid value),
   -- 'Originating Virtual Address' absent
   'Destination Virtual Address' = D3,
   'Destination Options' = (absent or a list of valid options),
   -- 'Data Options' absent
   TRANSMIT PORT (IUT-TD hub WebSocket)
   Address-Resolution-ACK,
   'Message ID' = M1,
   'Originating Virtual Address' = D3,
   -- 'Destination Virtual Address' absent
   -- 'Destination Options' absent
   -- 'Data Options' absent
   'Payload'
   'WebSocket-URIs' (W: a WebSocket URI with an invalid scheme)
   }
3. CHECK(that the IUT does not attempt to open a WebSocket to the invalid WebSocket-URI)

14.YY.3.3.2.3 Rejection of Invalid Certificate Outgoing Connection Test

Reference: YY.7.4

Purpose: To verify that the IUT will drop initiated connection attempts if the peer's certificate 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 invalid certificate during the connection. Verify that the WebSocket is not established.

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

1. MAKE(the IUT attempt to establish a direct connection to D3)
2. CHECK(that the IUT initiated a WebSocket connection)
3. CHECK(that the WebSocket connection was failed by the IUT)

[Add into clause 10.2, tests 10.2.X3, 10.2.X4, 10.2.X5, 10.2.X6]

10 NETWORK LAYER PROTOCOL TESTS

10.2 Router Functionality Tests

10.2.X3 Data Attributes Forwarding Test

Reference: 6.5, 6.5.4

Purpose: To verify that routers which connect multiple network supporting data attributes, correctly route data attributes.

Test Concept: With the IUT configured as a router between two networks which supports data attributes (such as BACnet/SC), send to the router a message which needs to be routed to the next network, which contains data attributes. Verify that the message is correctly routed and the data attributes are included in the routed message.

Configuration Requirements: The IUT shall configured as a router between 2 networks which support data attributes. If the IUT does not support this configuration this test shall be skipped.

1. TRANSMIT PORT A,
   DA = LOCAL BROADCAST,
   SOURCE = D1A,
   'Data Options' = (DATA_OPTS: a valid list of options),
   DNET = GLOBAL BROADCAST,
   DLEN = 0,
   Hop Count = 255,
   BACnet-Unconfirmed-Request-PDU,
   'Service Choice' = Who-Is
2. RECEIVE PORT B,
   DA = LOCAL BROADCAST,
   SA = IUT,
   'Data Options' = DATA_OPTS,
   DNET = GLOBAL BROADCAST,
   DLEN = 0,
   SNET = 1,
   SADR = D1A,
   Hop Count = (any integer x: 0 < x < 255),
   BACnet-Unconfirmed-Request-PDU,
   'Service Choice' = Who-Is

10.2.X4 Data Attributes Dropping Test

Reference: YY.2.3

Purpose: To verify that the IUT drops data_attributes from a received message before routing it to a network which does not support data_attributes.

Test Concept: With the IUT configured as a router from a network which support data_attributes (such as BACnet/SC) to a network which does not support data_attributes (such as BACnet/IP), send to the router a message which needs to be routed to the next network, and which contains data_attributes. Verify that message is correctly routed and the data_attributes are silently dropped.

1. TRANSMIT PORT A,
   DA = LOCAL BROADCAST,
   SOURCE = D1A,
   'Data Options' = (DATA_OPTS: a valid list of options),
   DNET = GLOBAL BROADCAST,
   DLEN = 0,
   Hop Count = 255,
   BACnet-Unconfirmed-Request-PDU,
   'Service Choice' = Who-Is
2. RECEIVE PORT B,
   DA = LOCAL BROADCAST,
   SA = IUT,
   DNET = GLOBAL BROADCAST,
   DLEN = 0,
   SNET = 1,
   SADR = D1A,
   Hop Count = (any integer x: 0 < x < 255),
   BACnet-Unconfirmed-Request-PDU,
   'Service Choice' = Who-Is

10.2.X5 Secure Path Test

Reference: YY.2.3.1

Purpose: To verify that a BACnet/SC to BACnet/SC router correctly conveys the 'Secure Path' header option on routed messages.

Test Concept: With the IUT configured as a router from BACnet/SC to BACnet/SC or another datalink which supports the 'Secure Path' header option, send to the router a message which needs to be routed to the next network, and which contains the Secure Path header option. Verify that Secure Path option is included in the routed message.

Configuration Requirements: The IUT is connected to 2 network which support data attributes and are secure. If the IUT does not support this configuration then this test shall be skipped.

1. TRANSMIT PORT A,
   DA = LOCAL BROADCAST,
   SOURCE = D1A,
   'Data Options' = (DATA_OPTS: secure path),
   DNET = GLOBAL BROADCAST,
   DLEN = 0,
   Hop Count = 255,
   BACnet-Unconfirmed-Request-PDU,
   'Service Choice' = Who-Is
2. RECEIVE PORT B,
   DA = LOCAL BROADCAST,
   SA = IUT,
   'Data Options' = (DATA_OPTS: secure path),
   DNET = GLOBAL BROADCAST,
   DLEN = 0,
   SNET = 1,
   SADR = D1A,
   Hop Count = (any integer x: 0 < x < 255),
   BACnet-Unconfirmed-Request-PDU,
   'Service Choice' = Who-Is

10.2.X6 Insecure Path Test

Reference: YY.2.3.1

Purpose: To verify that a non-BACnet/SC to BACnet/SC router does not include the 'Secure Path' header option on routed messages from the non-BACnet/SC network.

Test Concept: With the IUT configured as a router from BACnet/SC to a non-BACnet/SC which does not data options, send to the router a message on the non-BACnet/SC network which needs to be routed to the BACnet/SC network. Verify that the Secure Path option is not included in the routed message.

Configuration Requirements: The IUT is connected to 2 networks with PORT A connected to a network type which does not support secure path, and PORT B connected to a network type which does. If the IUT does not support this configuration then this test shall be skipped.

Test Steps:

1. TRANSMIT PORT A,
   DA = LOCAL BROADCAST,
   SOURCE = D1A,
   DNET = GLOBAL BROADCAST,
   DLEN = 0,
   Hop Count = 255,
   BACnet-Unconfirmed-Request-PDU,
   'Service Choice' = Who-Is
2. RECEIVE PORT B,
   DA = LOCAL BROADCAST,
   SA = IUT,
   -- 'Data Options' absent
   DNET = GLOBAL BROADCAST,
   DLEN = 0,
   SNET = 1,
   SADR = D1A,
   Hop Count = (any integer x: 0 < x < 255),
   BACnet-Unconfirmed-Request-PDU,
   'Service Choice' = Who-Is