BACnet® TESTING LABORATORIES

INTERIM TEST SPECIFICATION

To Be Used with Test Package 23.1
Version 3
September 17, 2023

Approved by the BTL Working Group September 14, 2023
Approved by the BTL Working Group Voting Members September 29 2023.
Published on October 1, 2023.

BACnet is a registered trademark of ASHRAE. ASHRAE does not endorse, approve or test products for compliance with ASHRAE standards. Compliance of listed products to the requirements of ASHRAE Standard 135 is the responsibility of BACnet International. BTL is a registered trademark of BACnet International.

Foreward

The purpose of this document is to define interim tests and other test package changes made to support testing of a device that supports functionality currently not covered in the released BTL Test Package. This document shall be applied and used with BTL Test Package 23.0.

Vendors who are planning to submit a device for testing and who implement Protocol_Revision 24 and higher, or which contain functionality not covered by the Official Test Package, should use this Interim Test document.

Please note that if the device contains functionality not yet covered by the official Test Package, nor by the Interim Tests document, development of new tests may be required for your device. Please contact the BTL Manager before submitting your device for testing to ensure you are aware of all tests that will need to be applied to your device.

The changes in this document are for interim use only and may or may not be used as documented here when the final changes are applied to the next Test Package revision. Devices tested using this interim test document shall be recalled for updated testing when the next revision of test package is released that includes the topics covered here.

In the following document, language to be added to existing clauses of ANSI/ASHRAE 135.1-2019 or any part of the Test Package 23.0 are indicated through the use of *italics*, while deletions are indicated by ~~strikethrough~~. Where entirely new sections are proposed to be added, plain type is used throughout.

© 2023 by BACnet International. All rights reserved.

Table of Contents

VERSIONS 3

BTL TEST PLAN CHANGES 4

3.56 Network Port Object 5
3.61 File Object 8
9.9 Data Link Layer – Secure Connect 10

BTL SPECIFIED TESTS CHANGES 14

7.3.2.X62.4.1 Valid Hierarchy Test 14
7.3.2.X62.3.9 No Commands if Changes_Pending Test 14
7.3.2.X62.3.X.1 GENERATE_CSR_FILE Command Test 15
7.3.2.X62.3.X.2 GENERATE_CSR_FILE Command Failure Test 16
7.3.2.X62.3.X.3 VALIDATE_CHANGES Command Test 16
7.3.2.X62.3.X.4 VALIDATE_CHANGES Command Failure Test 17
14.YY.1.1.16 Heartbeat-Request Initiation Test 17
14.YY.1.1.17 Configurable Reconnect Timeout Test 18
14.YY.1.1.X1 Node Heartbeat-Request Execution Test 19
14.YY.1.2.5 Connect-Request Response Wait Time Test 20
14.YY.1.2.X1 Node Heartbeat-Request Initialization Failure Test 20
14.YY.2.1.X1 SC_Hub_Function_Enable Property Test 21
14.YY.2.2.2 Connect-Request Wait Time Test 23
14.YY.2.2.X1 Hub Heartbeat-Request Initialization Failure Test 24
14.YY.3.2.2.1 Connect-Request Wait Time Test 25
19.Y.3.1 Adding a New Issuer Certificate to the Device 25
19.Y.3.2 Replace the Operational Certificate 27
19.Y.3.3 Removing an Outdated Issuer Certificate from the Device 28

© 2023 by BACnet International. All rights reserved.

Versions

Revision	Release	Changes
23.1 V1	2023-06-20	BTLWG-1372 - 135-2020cc-1. Update the Network Port Object and Add BACnet/SC Configuration Support BTLWG-1429 - 135-2020cc. Hub specific Tests and New tests for Annex AB Updates
23.1 V2	2023-09-05	BTLWG-1451 - Interim Tests Checklist document
23.1 V3	2023-09-??	Removed redundant Checklists sections. Added 'Test Steps' to Tests 7.3.2.X62.4.1, 7.3.2.X62.3.9, 7.3.2.X62.3.X2, 7.3.2.X62.3.X.3, 7.3.2.X62.3.X.4 14.YY.1.1.16 – Step 1, Removed strikethrough of '}' BTLWG-1469 - 14.YY.2.1.X1 - IUT no longer connected

© 2023 by BACnet International. All rights reserved.

BTL Test Plan Changes

This section of the document contains interim changes to the BTL Test Plan document to support testing of products with functionality outside the scope of the official test plan.

This section is ordered the same as the BTL Checklist and BTL Test Plan documents to allow easy navigation of the material.

All test changes can be found in the next major section.

© 2023 by BACnet International. All rights reserved.

3.56 Network Port Object

JIRA: BTLWG-1372 and BTLWG-1429

The current test package does not include Addendum cc changes to the Network Port object.

BTL Test Plan Changes

3.56.4 Supports Non-hierarchical Network Port Objects

The IUT contains, or can be made to contain, non-hierarchical Network Port objects.

Verify EPICS		
	Test Conditionality	*Must be executed.*
	Test Directives	*Verify IUT contains only Network Port objects with Protocol_Level equal to BACNET_APPLICATION or NON_BACNET_APPLICATION.*
	Testing Hints	
Verify EPICS		
	Test Conditionality	*Must be executed.*
	Test Directives	*Verify the Network Port object, Reference_Port and Additional_Reference_Ports properties are absent.*
	Testing Hints	
Verify EPICS		
	Test Conditionality	*Must be executed if the IUT claims Protocol_Revision >= 24.*
	Test Directives	*Verify each Network Port object contains all required properties based on its Network_Type.*
	Testing Hints	
Verify EPICS		
	Test Conditionality	*Must be executed if the IUT claims Protocol_Revision >= 24.*
	Test Directives	*Verify each Network Port object contains only valid optional properties based on its Network_Type.*
	Testing Hints	

3.56.4.5 Supports Hierarchical Network Port Objects

The IUT contains, or can be made to contain, a set of Network Port objects which form a hierarchy of protocols.

Verify EPICS		
	Test Conditionality	*Must be executed if the IUT claims Protocol_Revision >= 24.*
	Test Directives	*Verify the IUT contains a Network Port object for each Protocol_Level based on the Network_Type.*
	Testing Hints	
Verify EPICS		
	Test Conditionality	*Must be executed if the IUT claims Protocol_Revision >= 24.*
	Test Directives	*Verify that each Network Port object contains only required and optional properties based on its Network_Type and Protocol_Level.*
	Testing Hints	
BTL - 7.3.2.X62.4.1 - Valid Hierarchy Test		
	Test Conditionality	Must be executed.
	Test Directives	Repeat for each supported Network_Type at the BACNET_APPLICATION level.
	Testing Hints	

© 2023 by BACnet International. All rights reserved.

BTL - 7.3.2.X62.4.2 - Properties in Referenced Network Port Reflected in Top Network Port Object		
	Test Conditionality	Must be executed *if the IUT claims Protocol_Revision < 24.*
	Test Directives	Repeat for each supported Network_Type at the BACNET_APPLICATION level.
	Testing Hints	The test is written such that it tests all configured BACNET_APPLICATION Network Port objects so configuring the IUT to contain an example of each will allow the test to be run fewer times.
BTL - 7.3.2.X62.4.3 - Changes Reflected in Top Network Port Object		
	Test Conditionality	*Must be executed if the IUT claims Protocol_Revision < 24 and supports writable* ~~Test shall be skipped if the IUT does not support any writable~~ properties in its Network Port hierarchies.
	Test Directives	Repeat for each supported Network_Type at the BACNET_APPLICATION level.
	Testing Hints	The test is written such that it tests all configured BACNET_APPLICATION Network Port objects so configuring the IUT to contain an example of each will allow the test to be run fewer times.
BTL - 7.3.2.X62.4.4 - Changes Reflected in Lower Network Port Objects		
	Test Conditionality	*Must be executed if the IUT claims Protocol_Revision < 24 and supports writable* ~~Test shall be skipped if the IUT does not support any writable~~.
	Test Directives	Repeat for each supported Network_Type at the BACNET_APPLICATION level.
	Testing Hints	The test is written such that it tests all configured BACNET_APPLICATION Network Port objects so configuring the IUT to contain an example of each will allow the test to be run fewer times.

3.56~~.5~~.*6*Supports the Command Property

…

3.56~~.6~~.*7*Supports the DISCARD_CHANGES Command

…

3.56~~.7~~.8 Supports the RENEW_FD_REGISTRATION Command

…

3.56~~.8~~.*9*Supports the RESTART_SLAVE_DISCOVERY Command

…

3.56~~.9~~.*10* Supports the RENEW_DHCP Command

…

3.56~~.10~~.*11* Supports the RESTART_AUTONEGOTIATION Command

…

© 2023 by BACnet International. All rights reserved.

3.56.~~11~~.*12* Supports the DISCONNECT Command

…

3.56.~~12~~.*13* Supports the RESTART_PORT Command

…

3.56.14 Supports the GENERATE_CSR_FILE Command

The IUT supports the GENERATE_CSR_FILE Command in Network Port objects.

BTL - 7.3.2.X62.3.X.1 - GENERATE_CSR_FILE Command Test		
	Test Conditionality	*Must be executed if the IUT supports the GENERATE_CSR_FILE command.*
	Test Directives	
	Testing Hints	
BTL - 7.3.2.X62.3.X.2 - GENERATE_CSR_FILE Command Failure Test		
	Test Conditionality	*Must be executed if the IUT supports a Network Port object for which GENERATE_CSR_FILE is not supported.*
	Test Directives	
	Testing Hints	

3.56.15 Supports the VALIDATE_CHANGES Command

The IUT supports the VALIDATE_CHANGES command in Network Port objects.

BTL - 7.3.2.X62.3.X.3 - VALIDATE_CHANGES Command Test		
	Test Conditionality	*Must be executed if the IUT supports the VALIDATE_CHANGES command.*
	Test Directives	
	Testing Hints	
BTL - 7.3.2.X62.3.X.4 - VALIDATE_CHANGES Command Failure Test		
	Test Conditionality	*Must be executed if the IUT supports a Network Port object for which VALIDATE_CHANGES is not supported.*
	Test Directives	
	Testing Hints	

3.56.~~13~~.*16* Supports the Routing_Table Property

…

© 2023 by BACnet International. All rights reserved.

3.61 File Object

JIRA: BTLWG-1372

The current test package does not include Addendum cc changes to the File object.

BTL Test Plan Changes

3.61.3 Supports BACnet/SC Certificate Exchange

The IUT supports BACnet/SC Certificate Exchange using AtomicReadFile and AtomicWriteFile requests.

Verify Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims support for section 9.9.8 Supports Procedure to Replace BACnet/SC Certificates.
	Testing Hints	

~~3.61.3~~*3.61.4* Supports a Record-Based File Object for a Purpose Other Than Backup and Restore *or BACnet/SC Certificate Exchange*

For a device which contains a record-based File object for a purpose other than Backup and Restore *or BACnet/SC Certificate Exchange*, there are no testing requirements.

~~3.61.4~~*3.61.5* Supports a Stream-Based File Object for a Purpose Other Than Backup and Restore *or BACnet/SC Certificate Exchange*

The IUT supports a data stream-based File that is not accessed during Backup and Restore *or BACnet/SC Certificate Exchange*.

…

~~3.6.5~~*3.6.6* Contains a Writable Stream-Based File for a Purpose Other Than Backup and Restore *or BACnet/SC Certificate Exchange*

The IUT supports a data stream-based File that is not accessed during Backup and Restore *or BACnet/SC Certificate Exchange*.

BTL - 9.13.1.2.1 - Writing an Entire Stream-Based File		
	Test Conditionality	Must be executed.
	Test Directives	Apply to a file not used for Backup & Restore *or BACnet/SC Certificate Exchange*.
	Testing Hints	
BTL - 9.13.1.2.3 - Appending Data to the End of a File		
	Test Conditionality	If the file size cannot be changed or if the IUT does not support files that cannot be modified except by replacing the entire file, then this test shall be skipped.
	Test Directives	Apply to a file not used for Backup & Restore *or BACnet/SC Certificate Exchange*.
	Testing Hints	
135.1-2019 - 9.13.1.2.4 - Truncating a File		

© 2023 by BACnet International. All rights reserved.

	Test Conditionality	If the only value that the IUT accepts when writing File_Size is zero, then this test shall be skipped.
	Test Directives	Apply to a file not used for Backup & Restore *or BACnet/SC Certificate Exchange*.
	Testing Hints	
135.1-2019 - 9.13.1.2.5 - Deleting a File		
	Test Conditionality	If the file size cannot be changed, then this test shall be skipped.
	Test Directives	Apply to a file not used for Backup & Restore *or BACnet/SC Certificate Exchange*.
	Testing Hints	
135.1-2019 - 9.13.2.2.1 - Writing to a Stream Access File using Record Access		
	Test Conditionality	Must be executed.
	Test Directives	Apply to a file not used for Backup & Restore *or BACnet/SC Certificate Exchange*.
	Testing Hints	
135.1-2019 - 9.13.2.2.2 - Writing to a File with an Invalid Starting Position		
	Test Conditionality	Must be executed.
	Test Directives	Apply to a file not used for Backup & Restore *or BACnet/SC Certificate Exchange*.
	Testing Hints	
135.1-2019 - 9.13.2.2.4 - Writing to a Nonexistent File		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	

© 2023 by BACnet International. All rights reserved.

9.9 Data Link Layer – Secure Connect

JIRA: BTLWG-1372 and BTLWG-1429

The current test package does not include Addendum cc changes to the DLL Secure Connect.

BTL Test Plan Changes

9.9.1 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.~~ *If the IUT claims Protocol_Revision 24 or greater, this test must be executed. If the IUT claims Protocol_Revision 23 or lower and 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	

© 2023 by BACnet International. All rights reserved.

	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*~~ *a later version than HTTP1.1*
	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	
BTL - 7.3.2.X62.1.2 - Verify Network Configuration Through Network Port Object Test		
	Test Conditionality	If the device claims Protocol_Revision 16 or lower, this test shall be skipped.
	Test Directives	~~Perform at least once.~~ ~~Repeat each time the network is reconfigured for a test.~~ *Execute this test at least once on each Network Port object that has Network_Type = SECURE_CONNECT.*
	Testing Hints	

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

…

BTL - 14.YY.1.3.3 - Factory Defaults Test		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.1.X1 - Node Heartbeat-Request Execution Test		
	Test Conditionality	If the IUT supports Protocol_Revision 24 or greater, this test must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.1.2.X1 - Node Heartbeat-Request Initialization Failure Test		
	Test Conditionality	If the IUT supports Protocol_Revision 24 or greater, this test must be executed.
	Test Directives	

© 2023 by BACnet International. All rights reserved.

	Testing Hints	

9.9.3 Is Able to Operate as a Hub

…

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	
BTL - 14.YY.2.1.X1 - SC_Hub_Function_Enable Property Test		
	Test Conditionality	If the IUT supports Protocol_Revision 24 or greater, this test must be executed.
	Test Directives	
	Testing Hints	
BTL - 14.YY.2.2.X1 - Hub Heartbeat-Request Initialization Failure Test		
	Test Conditionality	If the IUT supports Protocol_Revision 24 or greater, this test must be executed.
	Test Directives	
	Testing Hints	

9.9.7 Supports Configuration Through Network Port Object

The IUT supports full, or partial, configuration of the data link through the Network Port object~~. Specifically, at least 1 property in the Network Port object which changes the behavior of the data link is writable.~~

BTL - 7.3.2.X62.1.1 - Configure Network Through Network Port Object Test		
	Test Conditionality	Must be executed.
	Test Directives	~~Perform at least once.~~ ~~Repeat each time the network is reconfigured for a test.~~ *Execute this test at least once on each Network Port object that has Network_Type = SECURE_CONNECT.*
	Testing Hints	

9.9.8 Supports Procedure to Replace BACnet/SC Certificates

The IUT supports full Procedure to Replace BACnet/SC Certificates. BACnet/SC certificates will usually have a finite lifetime. A procedure to replace expiring certificates is required to ensure the network is minimally interrupted during this transition.

Verify Checklist		
	Test Conditionality	Must be executed.
	Test Directives	Verify that the IUT claims support for section 3.61.3 Supports BACnet/SC Certificate Exchange
	Testing Hints	
BTL - 19.Y.3.1 Adding a New Issuer Certificate to the Device		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 19.Y.3.2 Replace the Operational Certificate		

© 2023 by BACnet International. All rights reserved.

	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	
BTL - 19.Y.3.3 Removing an Outdated Issuer Certificate from the Device		
	Test Conditionality	Must be executed.
	Test Directives	
	Testing Hints	

© 2023 by BACnet International. All rights reserved.

BTL Specified Tests Changes

This section contains all of the new and changed tests required by the interim test BTL Checklist and BTL Test Plan changes.

JIRA: BTLWG-1372 and BTLWG-1429

[Modify 7.3.2.X62.4.1]
7.3.2.X62.4.1 Valid Hierarchy Test
Reason for Change: New test per Addendum 135-2012ai *and modified per Addendum 135-2020cc-1.*

Purpose: To verify that the set of network port objects in the IUT are organized in a valid hierarchy.

Test Concept: Visit each Network Port object which represents a configured application layer port. Ensure that the top Network Port object has a Protocol_Level of BACNET_APPLICATION or NON_BACNET_APPLICATION. Visit each Network Port object in the hierarchy ensuring that the Protocol_Level properties are valid.

Test Steps:

1. REPEAT NP = (object id of each *hierarchical* Network Port object which has a Protocol_Level of
BACNET_APPLICATION or NON_BACNET_APPLICATION) {
~~2.~~ REPEAT NPx = (object id of each Network Port object , *Reference_Port* in NP's hierarchy) {
PL = READ (Network Port, NPx), Protocol_Level
IF PL is BACNET_APPLICATION or NON_BACNET_APPLICATION THEN
ERROR Invalid Protocol_Level in child Network Port object
IF PL is PHYSICAL THEN
VERIFY (Network Port, NPx), Reference_Port = 4194303
}
IF (Protocol_Revision >= 24 and Additional_Reference_Ports is present) THEN
IF (NP, Reference_Port property is not present) THEN
ERROR missing Reference_Port property
REPEAT (for each entry Network Port object, Additional_Reference_Ports) DO {
REPEAT NPx = (object id of each Network Port object, Additional_Reference_Ports in NP's hierarchy) DO {
PL = READ (Network Port, NPx), Protocol_Level
IF PL is BACNET_APPLICATION or NON_BACNET_APPLICATION THEN
ERROR Invalid Protocol_Level in child Network Port object
IF PL is PHYSICAL THEN
VERIFY (Network Port, NPx), Additional_Reference_Ports = (empty list)
}
}

[Modify 7.3.2.X62.3.9]
7.3.2.X62.3.9 No Commands if Changes_Pending Test
Reason for Change: New test per Addendum 135-2012ai *and modified per Addendum 135-2020cc-1.*

Purpose: To verify that the Network Port disallows commands, except DISCARD_CHANGES *and VALIDATE_CHANGES*, when Changes_Pending.

Test Concept: using Network Port object NP, write values to one or more properties, P1 .. Px, which utilize the pending changes functionality. Write each of the other commands and verify they are rejected.

Configuration Requirements: Execute the test on a Network Port object which supports the Command property.

© 2023 by BACnet International. All rights reserved.

Test Steps:

-- write some properties
1. REPEAT P = (P1 .. Px) {
    WRITE NP, P = (any valid value)
}

-- verify that changes are pending
2. VERIFY Changes_Pending = TRUE

-- write each supported Command value, except DISCARD_CHANGES *and VALIDATE_CHANGES*
3. REPEAT CMD = (all ~~non-IDLE~~ valid values that NP supports except DISCARD_CHANGES *and VALIDATE_CHANGES*) {
    TRANSMIT WriteProperty-Request
        'Object Identifier' = NP
        'Property' = Command,
        'Property Value' = CMD
    RECEIVE BACnet-Error-PDU
        'Error Class' = PROPERTY,
        'Error Code' = INVALID_VALUE_IN_THIS_STATE
}

-- revert the Network Port object
4. IF the IUT supports DISCARD_CHANGES THEN {
    WRITE Command = DISCARD_CHANGES
  } ELSE {
    MAKE (the IUT discard its changes)
  }

[Add 7.3.2.X62.3.X.1]
7.3.2.X62.3.X.1 GENERATE_CSR_FILE Command Test
Reason for Change: New test per Addendum 135-2020cc-1.

Purpose: To verify that the Network Port attempts to generate a new csr_file when commanded to.

Test Concept: Starting with a Network Port object which supports the GENERATE_CSR_FILE command. The port is commanded to GENERATE_CSR_FILE. The new certificate signing request file shall be referenced from the Certificate_Signing_Request_File property. The certificate signing request file shall be updated and a new private/public key pair shall be generated when the Command property is written to GENERATE_CSR_FILE. The certificate signing request file shall contain the X.509 'subject' distinguished name specified in the currently active operational certificate, and the public key from the new private/public key pair. If no operational certificate is currently active, then the X.509 'subject' distinguished name of the certificate signing request is a local matter but shall be globally unique.

Configuration Requirements: Execute the test on a Network Port object which supports the Command property and property Changes_Pending = FALSE
- CSR = This read-only property, of type BACnetObjectIdentifier, specifies the File object that contains the PKCS #10 certificate signing request as defined by RFC 5967 in PEM format.

Test Steps:

1. READ CSR = Certificate_Signing_Request_File
2. CHECK (CSR is non-empty BACnetObjectIdentifier referring to File object)
3. WHILE (the last read did not indicate 'End Of File') DO {
    TRANSMIT AtomicReadFile-Request,

© 2023 by BACnet International. All rights reserved.

‘Object Identifier’ = CSR,
‘File Start Position’ = (the next unread octet),
‘Requested Octet Count’ = MROC
RECEIVE AtomicReadFile-ACK,
'End Of File' = TRUE | FALSE,
‘File Start Position’ = (the next unread octet)
‘File Data’ = (CSR File contents of length MROC if 'End Of File' is FALSE
or of length MROC or less if 'End Of File' is TRUE)
}

4. TRANSMIT WriteProperty-Request,
'Object Identifier' = (the Network Port object),
'Property Identifier' = Command,
'Property Value' = GENERATE_CSR_FILE
5. RECEIVE BACnet-SimpleACK
6. VERIFY Changes_Pending = FALSE
7. WHILE (Command <> IDLE) DO {}
8. READ CSR = Certificate_Signing_Request_File
9. CHECK (CSR is non-empty BACnetObjectIdentifier referring to File object)
10. WHILE (the last read did not indicate 'End Of File') DO {
TRANSMIT AtomicReadFile-Request,
‘Object Identifier’ = CSR,
‘File Start Position’ = (the next unread octet),
‘Requested Octet Count’ = MROC
RECEIVE AtomicReadFile-ACK,
'End Of File' = TRUE | FALSE,
‘File Start Position’ = (the next unread octet)
‘File Data’ = (CSR File contents of length MROC if 'End Of File' is FALSE
or of length MROC or less if 'End Of File' is TRUE)
}
11. VERIFY (that a file content from Step 3 <> file content from Step 10)

[Add 7.3.2.X62.3.X.2]
7.3.2.X62.3.X.2 GENERATE_CSR_FILE Command Failure Test
Reason for Change: New test per Addendum 135-2020cc-1.

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

Test Concept: With a Network Port object for a network which does not support GENERATE_CSR_FILE. Verify that the correct error code is returned.

Configuration Requirements: property Changes_Pending = FALSE.

Test Steps:

1. TRANSMIT WriteProperty-Request,
'Object Identifier' = (the Network Port object),
'Property Identifier' = Command,
'Property Value' = GENERATE_CSR_FILE
2. RECEIVE BACnet-Error-PDU
'Error Class' = PROPERTY,
'Error Code' = OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED

[Add 7.3.2.X62.3.X.3]
7.3.2.X62.3.X.3 VALIDATE_CHANGES Command Test

© 2023 by BACnet International. All rights reserved.

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

Purpose: To verify that the Network Port attempts to perform the required validations on this property when commanded to.

Test Concept: Starting with a Network Port object which supports the VALIDATE_CHANGES command. The port is commanded to VALIDATE_CHANGES. This command shall initiate a validation of the values of the properties of this port as specified in each property. If a property is present but not used, based on the Network_Type, it shall not be validated. The value of the Command_Validation_Result property shall be updated to indicate the validation result.

Test Steps:

1. READ V1 = Command_Validation_Result
2. READ CP = Changes_Pending
   -- request a VALIDATE_CHANGES command, and wait for it to timeout
3. WRITE Command = VALIDATE_CHANGES
4. VERIFY Changes_Pending = CP
5. WHILE (Command <> IDLE) DO {}
6. VERIFY Command_Validation_Result = Any value different from V1

[Add 7.3.2.X62.3.X.4]
7.3.2.X62.3.X.4 VALIDATE_CHANGES Command Failure Test
Reason for Change: New test per Addendum 135-2020cc-1.

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

Test Concept: With a Network Port object for a network which does not support VALIDATE_CHANGES. Verify that the correct error code is returned.

Test Steps:

1. TRANSMIT WriteProperty-Request,
   'Object Identifier' = (the Network Port object),
   'Property Identifier' = Command,
   'Property Value' = VALIDATE_CHANGES
2. RECEIVE BACnet-Error-PDU
   'Error Class' = PROPERTY,
   'Error Code' = OPTIONAL_FUNCTIONALITY_NOT_SUPPORTED

[Modify 14.YY.1.1.16]
14.YY.1.1.16 Heartbeat-Request Initiation Test

Reference: ~~YY.2.14, YY.2.15, YY.6.3~~ 1 135-2020 Clause AB.6.3 and 135-2020 Addendum cc, Clause 12.56.Y10 and per CR-527

Purpose: To verify that the device 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. W*~~wait the IUT's configured heart-beat interval plus 10 seconds and verify that the IUT sent a Heartbeat-Request, ensuring that no BVLCs are sent to the IUT during that period. *If the IUT claims Protocol_Revision 24 or greater heartbeat interval is the Network Port object, SC_Heartbeat_Timeout property.*

© 2023 by BACnet International. All rights reserved.

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'41'}), -- Secure Path
        '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'41' or a list of valid header options including Secure Path}),
        'BACnet NPDU' =
            ReadProperty-ACK,
            'Object Identifier' = (the IUT's Device object),
            'Property Identifier' = Object_Name,
            'Property Value' = (the IUT's device object name)
    ~~*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*~~

2~~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

3~~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

[Modify 14.YY.1.1.17]

14.YY.1.1.17 Configurable Reconnect Timeout Test

Reference: ~~YY.6.1~~*135-2020 Clause AB.6.1 and 135-2020 Addendum cc, Clause 12.56.Y6 and 12.56.Y7.*

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

© 2023 by BACnet International. All rights reserved.

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*, minRT*, and no longer than *maxRT*~~600~~ seconds before attempting to reconnect. *If the IUT claims Protocol_Revision 23 or lower, minRT is a configurable parameter within the IUT and maxRT is fixed at 600 seconds. If the IUT claims Protocol_Revision 24 or greater, minRT is the Network Port object, SC_Minimum_Reconnect_Time property and maxRT is the SC_Maximum_Reconnect_Time property.*

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, *minRT*~~RT~~, within the range supported by the IUT and within 2 .. 300 seconds *and maxRT, within the range supported by the IUT and within 2 .. 600 seconds*. The IUT starts the test *disconnected from the TD*~~powered off~~. ~~If the IUT has a fixed reconnect timeout, this test shall be skipped.~~

Test Steps:

1. MAKE(place the TD in a mode where it will not accept a websocket connection~~incoming connections~~)
2~~1~~. MAKE(the IUT *attempt to* connect to the TD)
3. T1 = Local Time
~~2. CHECK(that the IUT attempts to open a new WebSocket with the TD)~~
4~~3~~. MAKE(place the TD in a mode where it will accept incoming connections)
~~4. WAIT *minRT*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)
7. T2 = Local Time
8. CHECK (T2 - T1 >= minRT)
9. CHECK (T2 - T1 <= maxRT)

[Add 14.YY.1.1.X1]
14.YY.1.1.X1 Node Heartbeat-Request Execution Test

Reference: Addendum cc Clause AB.5.3.1.

Purpose: To verify that a node device accepts and responds to Heartbeat-Requests.

© 2023 by BACnet International. All rights reserved.

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

Configuration Requirements: The IUT is configured as a node and connected to the TD.

Test Steps:

1. MAKE(the TD generate a Heartbeat-Request)
2. RECEIVE PORT (TD-IUT hub WebSocket),
   Heartbeat-ACK,
   '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

[Modify 14.YY.1.2.5]

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

Reference: ~~YY.6.2, YY.7.5.5~~ 135-2020 Addendum cc, Clause 12.56.Y8

Purpose: To verify that the device will close the WebSocket if a response to a Connect-Request is not received before the connection wait timer expires. *If the IUT claims Protocol_Revision 24 or greater connect wait timeout is the Network Port object, SC_Connect_Wait_Timeout property.*

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 connect wait timeout
5. CHECK(that the IUT closed the WebSocket)

[Add 14.YY.1.2.X1]

14.YY.1.2.X1 Node Heartbeat-Request Initialization Failure Test

Reference: Addendum cc Clause AB.6.3.

Purpose: To verify that a Node will disconnect if a Heartbeat-ACK is not received.

Test Concept: With the IUT connected to the TD as the primary hub, allow the IUT to connect to the TD. OD sends a ReadProperty request to the IUT every HB / 2 seconds. HB is the value of the IUTs SC_Heartbeat_Timeout property. Verify that the IUT does not initiate a Heartbeat-Request. Stop sending messages to the IUT. Wait HB plus 10 seconds

© 2023 by BACnet International. All rights reserved.

and verify the IUT sends a Heartbeat-Request, times out waiting for a Heartbeat-ACK and then the IUT sends a Disconnect-Request.

Configuration Requirements: Configure the SC_Heartbeat_Timeout property of the TD to be 2 times HB. Place the TD in a mode where it will not respond to Heartbeat-Requests.

Test Steps:

```
1.      REPEAT N = (1..Z) {
                TRANSMIT Encapsulated-NPDU,
                        'Message ID' =                          (M: any valid value),
                        'Originating Virtual Address' =         (OD's VMAC),
                        -- 'Destination Virtual Address' absent
                        'Destination Options'                   (absent or any valid value),
                        'Data Options' =                        ({ X'41'}), -- Secure Path
                        'BACnet NPDU' =
                                ReadProperty-Request,
                                'Object Identifier' =           (the IUT's Device object),
                                'Property Identifier' =         Object_Name
                RECEIVE Encapsulated-NPDU,
                        'Message ID' =                          M,
                        -- 'Originating Virtual Address' absent
                        'Destination Virtual Address' =         (OD's VMAC),
                        'Destination Options'                   (absent or any valid value),
                        'Data Options' =                        ({ X'41' or a list of valid header options
including Secure Path}),
                        'BACnet NPDU' =
                                ReadProperty-ACK,
                                'Object Identifier' =           (the IUT's Device object),
                                'Property Identifier' =         Object_Name,
                                'Property Value' =              (the IUT's device object name)
                WAIT HB / 2
        }
2.      CHECK(that the IUT did not send a Heartbeat-Request during step 1)
-- Since we already waited ½ of HB, only HB / 2 of that interval is now given for the IUT to
-- generate a Heartbeat-Request
3.      BEFORE HB / 2 + 10s
                RECEIVE Heartbeat-Request,
                        'Message ID' =                          (M: any valid value)
                        -- 'Originating Virtual Address' absent
                        -- 'Destination Virtual Address' absent
                        'Destination Options' =                 (absent or any valid value),
                        -- 'Data Options' absent
4.      BEFORE 2 seconds
                RECEIVE Disconnect-Request,
                        'Message ID' =                          (M: any valid value)
                        -- 'Originating Virtual Address' absent
                        -- 'Destination Virtual Address' absent
                        'Destination Options' =                 (absent or any valid value),
                        -- 'Data Options' absent
```

[Add 14.YY.2.1.X1]

14.YY.2.1.X1 SC_Hub_Function_Enable Property Test

Reference: Addendum cc Clause 12.56.Y14.

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

© 2023 by BACnet International. All rights reserved.

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

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

Test Steps:

```
1.      MAKE(the TD open a WebSocket to the IUT's hub function)
2.      TRANSMIT PORT (TD-IUT hub WebSocket)
                Connect-Request,
                'Message ID' =                          (M1: any valid value),
                -- 'Originating Virtual Address' absent
                -- 'Destination Virtual Address' absent
                -- 'Destination Options' absent
                -- 'Data Options' absent
                'VMAC Address' =                        (TD's VMAC),
                'Device UUID' =                         (TD's UUID),
                'Maximum BVLC Length' =         (the TD's maximum BVLC accepted length),
                'Maximum NPDU Length' =         (the TD's maximum NPDU accepted length)
3.      RECEIVE PORT (TD-IUT hub WebSocket)
                Connect-Accept,
                'Message ID' =                          M1,
                -- 'Originating Virtual Address' absent
                -- 'Destination Virtual Address' absent
                'VMAC Address' =                        (IUT's VMAC),
                'Device UUID' =                         (IUT's UUID),
                'Maximum BVLC Length' =         (the IUT's maximum BVLC accepted length),
                'Maximum NPDU Length' =         (the IUT's maximum NPDU accepted length)
4.      IF (SC_Hub_Function_Enable is writable) THEN
                WRITE SC_Hub_Function_Enable = FALSE
                TRANSMIT ReinitializeDevice-Request
                        'Reinitialized State of Device' = WARMSTART | ACTIVATE_CHANGES
                        'Password' = (any valid password)
                RECEIVE BACnet-SimpleACK-PDU
        ELSE
                MAKE (SC_Hub_Function_Enable = FALSE)
5.      WAIT Activate Changes Fail Time
6.      CHECK(that the TD attempts and fails to open a WebSocket to the IUT)
7.      CHECK(that the IUT opens a WebSocket with the TD)
8.      TRANSMIT PORT (IUT-TD failover 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' = (IUT's VMAC),
                        'Device UUID' = (IUT's UUID),
                        'Maximum BVLC Length' = (the IUTs maximum BVLC accepted length),
                        'Maximum NPDU Length' = (the IUTs maximum NPDU accepted length)
9.      RECEIVE PORT (IUT-TD failover hub WebSocket)
                Connect-Accept,
                        'Message ID' = M2,
                        -- 'Originating Virtual Address' absent
                        -- 'Destination Virtual Address' absent
```


© 2023 by BACnet International. All rights reserved.

```
                    '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)
10.     VERIFY (SC_Hub_Function_Enable = FALSE)
11.     IF (SC_Hub_Function_Enable is writable) THEN
            WRITE SC_Hub_Function_Enable = TRUE
            TRANSMIT ReinitializeDevice-Request
                'Reinitialized State of Device' = WARMSTART | ACTIVATE_CHANGES
                'Password' = (any valid password)
            RECEIVE BACnet-SimpleACK-PDU
        ELSE
            MAKE (SC_Hub_Function_Enable = TRUE)
12.     WAIT Activate Changes Fail Time
13.     MAKE(the TD open a WebSocket to the IUT's hub function)
14.     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)
15.     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)
```

[Modify 14.YY.2.2.2]

14.YY.2.2.2 Connect-Request Wait Time Test

Reference: ~~YY.6.2, YY.7.5.5~~ 135-2020 Addendum cc, Clause 12.56.Y8

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. *If the IUT claims Protocol_Revision 24 or greater connect wait timeout is the Network Port object, SC_Connect_Wait_Timeout property*.

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)

© 2023 by BACnet International. All rights reserved.

[Add 14.YY.2.2.X1]

14.YY.2.2.X1 Hub Heartbeat-Request Initialization Failure Test

Reference: Addendum cc Clause AB.5.3.1.

Purpose: To verify that a Hub initiates a Heartbeat-Request before attempting to terminate a connection.

Test Concept: With the IUT operating as a hub, the TD connects to the IUT. The TD sends a ReadProperty request to the IUT every HB / 2 seconds. HB is the value of the IUTs SC_Heartbeat_Timeout property. Verify that the IUT does not initiate a Heartbeat-Request. Stop sending messages to the IUT. Wait HB plus 10 seconds and verify the IUT sends a Heartbeat-Request, times out waiting for a Heartbeat-ACK and then the IUT sends a Disconnect-Request.

Configuration Requirements: Configure the SC_Heartbeat_Timeout property of the TD to be 2 times HB. Place the TD in a mode where it will not respond to Heartbeat-Requests.

Test Steps:

```
1.      REPEAT N = (1..Z) {
                TRANSMIT Encapsulated-NPDU,
                        'Message ID' =                          (M: any valid value),
                        -- 'Originating Virtual Address' absent
                        'Destination Virtual Address' =         (IUT's VMAC),
                        'Destination Options'                   (absent or any valid value),
                        'Data Options' =                        ({ X'41'}), -- Secure Path
                        'BACnet NPDU' =
                                ReadProperty-Request,
                                'Object Identifier' =           (the IUT's Device object),
                                'Property Identifier' =         Object_Name
                RECEIVE Encapsulated-NPDU,
                        'Message ID' =                          M,
                        'Originating Virtual Address' =         (IUT's VMAC),
                        -- 'Destination Virtual Address' absent
                        'Destination Options'                   (absent or any valid value),
                        'Data Options' =                        ({ X'41' or a list of valid header options
including Secure Path}),
                        'BACnet NPDU' =
                                ReadProperty-ACK,
                                'Object Identifier' =           (the IUT's Device object),
                                'Property Identifier' =         Object_Name,
                                'Property Value' =              (the IUT's device object name)
                WAIT HB / 2
        }
2.      CHECK(that the IUT did not send a Heartbeat-Request during step 1)
-- Since we already waited ½ of HB, only HB / 2 of that interval is now given for the IUT to
-- generate a Heartbeat-Request
3.      BEFORE HB / 2 + 10s
                RECEIVE Heartbeat-Request,
                        'Message ID' =                          (M: any valid value)
                        -- 'Originating Virtual Address' absent
                        -- 'Destination Virtual Address' absent
                        'Destination Options' =                 (absent or any valid value),
                        -- 'Data Options' absent
4.      BEFORE 2 seconds
                RECEIVE Disconnect-Request,
                        'Message ID' =                          (M: any valid value)
                        -- 'Originating Virtual Address' absent
```


© 2023 by BACnet International. All rights reserved.

-- 'Destination Virtual Address' absent
'Destination Options' = (absent or any valid value),
-- 'Data Options' absent

[Modify 14.YY.3.2.2.1]
14.YY.3.2.2.1 Connect-Request Wait Time Test

Reference: ~~YY.6.2, YY.7.5.5~~ 135-2020 Addendum cc, Clause 12.56.Y8

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. *If the IUT claims Protocol_Revision 24 or greater connection wait timer is the Network Port object, SC_Connect_Wait_Timeout property.*

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

Test Steps:

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)

[Add 19.Y.3.1]
19.Y.3.1 Adding a New Issuer Certificate to the Device
New test per Addendum 135-2020cc-3.

Purpose: This procedure is used to add a new issuer certificate to the network and is used in conjunction with other procedures in order to ensure that the network remains accessible. As such, it is important to not remove or replace the currently used issuer certificate, otherwise the nodes will no longer be able to connect to the network due to certification validation failures when the issuer certificate that signed the operational certificate is missing

Test Concept:
In BACnet/SC Network Port objects there are two issuer certificate files. This procedure assumes that the network is already up and running and as such one of those files contains the currently active issuer certificate. The other issuer certificate file will either be empty or contain an outdated issuer certificate. The procedure will replace the content of this empty or outdated issuer certificate file.
To add the new issuer certificate:
1. Device A determines the File object that represents the currently used issuer certificate for this network by reading the file content of one of the issuer certificate files and comparing it to the known current issuer certificate.
2. Device A writes the new issuer certificate to the certificate file that represents the other issuer certificate.
3. Device A sends a ReinitializeDevice (ACTIVATE_CHANGES or WARMSTART, <password>) message to activate the changes to the Network Port object

Configuration Requirements: These procedures assume that the BACnet/SC device to be configured contain a Network Port object with Network_Type of SECURE_CONNECT, the BACnet/SC network is active, this node have a valid operational certificate. When performing the AtomicReadFile and AtomicWriteFile services, a Maximum Requested Octet Count (MROC) and a Maximum Write Data Length (MWDL) shall be calculated before starting the test. These values shall be used during the test. MROC shall be 16 octets less than the minimum of the TD's Max_APDU_Length_Accepted and the IUT's maximum transmittable APDU length. MWDL shall be 21 octets less than the minimum of the TD's maximum transmittable APDU length and the IUT's Max_APDU_Length_Accepted. (See 135-2020 Addendum cc, Clause 19.Y.3.1)
Define:
IC = Network Port object, Issuer_Certificate_File property. See 135-2020 Addendum cc, Clause 12.56.Y25.

© 2023 by BACnet International. All rights reserved.

Test Steps:

```
1.      READ IC =(Issuer_Certificate_Files 1, Issuer_Certificate_Files 2)
2.      CHECK (IC is non-empty of type BACnetARRAY[2] of BACnetObjectIdentifier referring to 2 File
        objects)
3.      WHILE (the last read did not indicate 'End Of File') DO {
                TRANSMIT AtomicReadFile-Request,
                        ‘Object Identifier’ = IC[1],
                        ‘File Start Position’ = (the next unread octet),
                        ‘Requested Octet Count’ = MROC
                RECEIVE AtomicReadFile-ACK,
                        'End Of File' = TRUE | FALSE,
                        ‘File Start Position’ = (the next unread octet)
                        ‘File Data’ = (IC[1] File contents of length MROC if 'End Of File' is FALSE
                                or of length MROC or less if 'End Of File' is TRUE)
4.      WHILE (the last read did not indicate 'End Of File') DO {
                TRANSMIT AtomicReadFile-Request,
                        ‘Object Identifier’ = IC[2],
                        ‘File Start Position’ = (the next unread octet),
                        ‘Requested Octet Count’ = MROC
                RECEIVE AtomicReadFile-ACK,
                        'End Of File' = TRUE | FALSE,
                        File Start Position’ = (the next unread octet)
                        ‘File Data’ = (IC[2] File contents of length MROC if 'End Of File' is FALSE
                                or of length MROC or less if 'End Of File' is TRUE)
                                }
5.      IF (IC[1] File contents is = the known current issuer certificate) THEN
                WRITE IC[2], File_Size = 0
                REPEAT Z = (0 through the file size, in increments of MWDL) DO {
                        TRANSMIT AtomicWriteFile-Request
                                File Identifier’ = IC[2]
                                ‘File Start Position’ = Z
                                ‘Record Data’ = (file contents of the new issuer certificate, the number of octets
                                        being the lesser of (file size - Z) and MWDL)
                                RECEIVE AtomicWriteFile-ACK
                                        ‘File Start Position’ = Z
                                }
        ELSE
                READ FS = IC[1], File_Size
                WRITE IC[1], File_Size = 0
                REPEAT Z = (0 through the file size, in increments of MWDL) DO {
                        TRANSMIT AtomicWriteFile-Request
                                ‘File Identifier’ = IC[1]
                                ‘File Start Position’ = Z
                                ‘Record Data’ = (file contents of the new issuer certificate, the number of octets
                                        being the lesser of (file size - Z) and MWDL)
                                RECEIVE AtomicWriteFile-ACK
                                        ‘File Start Position’ = Z
                                }


6.      TRANSMIT ReinitializeDevice-Request
                'Reinitialized State of Device' = WARMSTART | ACTIVATE_CHANGES
                'Password' = (any valid password)
7.      RECEIVE BACnet-SimpleACK-PDU
```


© 2023 by BACnet International. All rights reserved.

8. WAIT Activate Changes Fail Time
9. VERIFY Changes_Pending = FALSE
10. VERIFY the connection shall be re-established

[Add 19.Y.3.2]
19.Y.3.2 Replace the Operational Certificate
New test per *Addendum 135-2020cc-3.*

Purpose: This test case verifies that the IUT can execute a replacement of operational certificate

Test Concept:
1. If a new private/public key pair is required and supported, device A writes the GENERATE_CSR_FILE command to the Command property.
2. Device A uploads the File object that contains the CSR from the node.
3. Device A requests the signing CA generate a new operational certificate based on the CSR data from step 2.
4. Device A writes the File object that represents the operational certificate for this node.
5. Device A sends a ReinitializeDevice(ACTIVATE_CHANGES or WARMSTART, <password>) message to activate the changes to the Network Port object.

Configuration Requirements: These procedures assume that the BACnet/SC device to be configured contain a Network Port object with Network_Type of SECURE_CONNECT, the BACnet/SC network is active, this node have a valid operational certificate. When performing the AtomicReadFile and AtomicWriteFile services, a Maximum Requested Octet Count (MROC) and a Maximum Write Data Length (MWDL) shall be calculated before starting the test. These values shall be used during the test. MROC shall be 16 octets less than the minimum of the TD's Max_APDU_Length_Accepted and the IUT's maximum transmittable APDU length. MWDL shall be 21 octets less than the minimum of the TD's maximum transmittable APDU length and the IUT's Max_APDU_Length_Accepted. (See 135-2020 Addendum cc, Clause 19.Y.3.2)
Define:
CSR = Network Port object, Certificate_Signing_Request_File property. See 135-2020 Addendum cc, Clause 12.56.Y26
OC = Network Port object, *Operational_Certificate_File* property. See 135-2020 Addendum cc, Clause 12.56.Y24

Test Steps:

```
1.  IF (the IUT is capable of generating a certificate signing request file) THEN
            -- request the IUT generate a CSR, and wait for it to timeout
            WRITE Command = GENERATE_CSR_FILE
            WHILE (Command <> IDLE) DO {}
            ELSE
            MAKE (CSR File contains the PKCS #10 certificate signing request in PEM format)
2.  READ CSR = Certificate_Signing_Request_File
3.  READ OC = Operational_Certificate_File
4.  CHECK (CSR is non-empty BACnetObjectIdentifier referring to File object)
5.  WHILE (the last read did not indicate 'End Of File') DO {
                        TRANSMIT AtomicReadFile-Request,
                                'Object Identifier' = CSR,
                                'File Start Position' = (the next unread octet),
                                'Requested Octet Count' = MROC
                        RECEIVE AtomicReadFile-ACK,
                                'End Of File' = TRUE | FALSE,
                                'File Start Position' = (the next unread octet)
                                'File Data' = (CSR File contents of length MROC if 'End Of File' is FALSE
                                        or of length MROC or less if 'End Of File' is TRUE)
                        }
6.  Requests the signing CA generate a new operational certificate OC-Data based on the CSR CSR File
```


© 2023 by BACnet International. All rights reserved.

contentsdata from step 5

7. READ FS = OC, File_Size
    WRITE OC, File_Size = 0
    REPEAT Z = (0 through the file size, in increments of MWDL) DO {
        TRANSMIT AtomicWriteFile-Request
            'File Identifier' = OC
            'File Start Position' = Z
            'Record Data' = (file contents of the new operational certificate OC-Data, the number of octets
                being the lesser of (file size - Z) and MWDL)
        RECEIVE AtomicWriteFile-ACK
            'File Start Position' = Z
        }
4. TRANSMIT ReinitializeDevice-Request
    'Reinitialized State of Device' = WARMSTART | ACTIVATE_CHANGES
    'Password' = (any valid password)
5. RECEIVE BACnet-SimpleACK-PDU
7. WAIT Activate Changes Fail Time
8. VERIFY Changes_Pending = FALSE
9. VERIFY the connection shall be re-established

[Add 19.Y.3.3]

19.Y.3.3 Removing an Outdated Issuer Certificate from the Device

New test per Addendum 135-2020cc-3.

Purpose: This test case verifies that the IUT can execute a Removing an Issuer Certificate.

Test Concept:
1. Device A determines the File object that represents the currently used issuer certificate for this network by reading the file content of one of the issuer certificate files and comparing it to the known current issuer certificate.
2. Device A writes zero to the File_Size property of the issuer certificate File object that represents the other issuer certificate.
3. Device A sends a ReinitializeDevice(ACTIVATE_CHANGES or WARMSTART, <password>) message to activate the changes to the Network Port object.

Configuration Requirements: These procedures assume that the BACnet/SC device is configured to contain a Network Port object with Network_Type of SECURE_CONNECT, the BACnet/SC network is active, this node have a valid operational certificate. When performing the AtomicReadFile and AtomicWriteFile services, a Maximum Requested Octet Count (MROC) and a Maximum Write Data Length (MWDL) shall be calculated before starting the test. These values shall be used during the test. MROC shall be 16 octets less than the minimum of the TD's Max_APDU_Length_Accepted and the IUT's maximum transmittable APDU length. MWDL shall be 21 octets less than the minimum of the TD's maximum transmittable APDU length and the IUT's Max_APDU_Length_Accepted. (See 135-2020 Addendum cc, Clause 19.Y.3.3)
Define:
IC = Network Port object, Issuer_Certificate_File property. See 135-2020 Addendum cc, Clause 12.56.Y25.

Test Steps:

1. READ IC =(Issuer_Certificate_Files 1, Issuer_Certificate_Files 2)
2. CHECK (CSR is non-empty of type BACnetARRAY[2] of BACnetObjectIdentifier referring to 2 File objects)
3. WHILE (the last read did not indicate 'End Of File') DO {
        TRANSMIT AtomicReadFile-Request,
            'Object Identifier' = IC[1],
            'File Start Position' = (the next unread octet),

© 2023 by BACnet International. All rights reserved.

```
                ‘Requested Octet Count’ = MROC
        RECEIVE AtomicReadFile-ACK,
                'End Of File' = TRUE | FALSE,
                ‘File Start Position’ = (the next unread octet)
                ‘File Data’ = (IC[1] File contents of length MROC if 'End Of File' is FALSE
                                or of length MROC or less if 'End Of File' is TRUE)
4.      WHILE (the last read did not indicate 'End Of File') DO {
        TRANSMIT AtomicReadFile-Request,
                ‘Object Identifier’ = IC[2],
                ‘File Start Position’ = (the next unread octet),
                ‘Requested Octet Count’ = MROC
        RECEIVE AtomicReadFile-ACK,
                'End Of File' = TRUE | FALSE,
                File Start Position’ = (the next unread octet)
                ‘File Data’ = (IC[2] File contents of length MROC if 'End Of File' is FALSE
                                or of length MROC or less if 'End Of File' is TRUE)
                        }
5.      IF (IC[1] File contents is = the known current issuer certificate) THEN
        WRITE IC[2], File_Size = 0
        ELSE
        WRITE IC[1], File_Size = 0
6.      TRANSMIT ReinitializeDevice-Request
        'Reinitialized State of Device' = WARMSTART | ACTIVATE_CHANGES
        'Password' = (any valid password)
7.      RECEIVE BACnet-SimpleACK-PDU
8.      MAKE (change the TD network setup and the network setup of all other devices on the network to match
        the target network setup)
9.      WAIT Activate Changes Fail Time
10.     VERIFY Changes_Pending = FALSE
```


© 2023 by BACnet International. All rights reserved.