ONE WORLD ○ OUR APPROVAL

Wireless test report – 370566-1TRFWL

Applicant:

JMA Wireless (Teko Telecom Srl)

Product name:

TEKO XRAN RadioUnit

Model:	Model variant:
XR19AX35WM2/48Y	XRAF2335WM2/48Y

FCC ID:	
XM2-X19AX35M2	XM2-XAF2335M2

Specifications:

WINNF-TS-0122, Version V1.0.1 – CBSD requirements

Test and Certification for Citizens Broadband Radio Service (CBRS); Conformance and Performance Test Technical Specification; CBSD/DP as Unit Under Test (UUT)

WINNF-IN-00129, Version V1.0.0.0

WInnForum CBSD/DP UUT Security Test 6 Cases Tutorial

Date of issue: June 25, 2019

Test engineer(s):	Andrey Adelberg, Senior Wireless/EMC Specialist	Signature:	Andrey Adelberg
Reviewed by:	Tom Tidwell	Signature:	

Test location(s)

Company name	Nemko Canada Inc.
Address	303 River Road
City	Ottawa
Province	Ontario
Postal code	K1V 1H2
Country	Canada
Telephone	+1 613 737 9680
Facsimile	+1 613 737 9691
Toll free	+1 800 563 6336
Website	www.nemko.com
Site number	FCC: CA2040; IC: 2040A-4 (3 m SAC)

Limits of responsibility

Note that the results contained in this report relate only to the items tested and were obtained in the period between the date of initial receipt of samples and the date of issue of the report.
This test report has been completed in accordance with the requirements of ISO/IEC 17025. All results contain in this report are within Nemko Canada's ISO/IEC 17025 accreditation.

Copyright notification

Table of contents

Section 1. Report summary

1.1 Applicant and manufacturer

Company name	JMA Wireless (Teko Telecom Srl)
Address	Via Antonio Meucci, 24
City	Castel San Pietro Terme
Province/State	BO
Postal/Zip code	40024
Country	Italy

1.2 Test specifications

WINNF-TS-0122 Version V1.0.1 (28 September 2018)	Test and Certification for Citizens Broadband Radio Service (CBRS); Conformance and Performance Test Technical Specification; CBSD/DP as Unit Under Test (UUT)
WINNF-IN-00129, Version V1.0.0.0	WInnForum CBSD/DP UUT Security Test 6 Cases Tutorial
FCC 47 CFR Part 96	Citizens Broadband Radio Service
WINNF-TS-0016 Version V1.2.1	Signaling Protocols and Procedures for Citizens Broadband Radio Service (CBRS): Spectrum Access System (SAS) - Citizens Broadband Radio Service Device (CBSD) Interface Technical Specification

1.3 Statement of compliance

In the configuration tested, the EUT was found compliant.

Testing was performed against all relevant requirements of the test standard except as noted in section 1.5 below. Results obtained indicate that the product under test complies in full with the requirements tested. The test results relate only to the items tested.

See "Summary of test results" for full details.

1.4 Exclusions

This test report covers only CBSD requirements.

1.5 Test report revision history

Revision #	Date of issue	Details of changes made to test report
TRF	June 25, 2019	Original report issued

Section 2. Summary of test results

2.1 WINNF-TS-0122 requirements test results

Table 2.1-1: *Domain Proxy requirements results*

Section	Test description	Verdict
6.1.4.1.1	Multi-Step registration	Pass
6.1.4.1.3	Single-Step registration for Cat A CBSD	Pass
6.1.4.1.7	Registration due to change of an installation parameter	Pass
6.1.4.2.1	Missing Required parameters (responseCode 102)	Pass
6.1.4.2.3	Pending registration (responseCode 200)	Pass
6.1.4.2.5	Invalid parameters (responseCode 103)	Pass
6.1.4.2.7	Blacklisted CBSD (responseCode 101)	Pass
6.1.4.2.9	Unsupported SAS protocol version responseCode 100)	Pass
6.1.4.2.11	Group Error (responseCode 201)	Pass
6.1.4.3.1	Category A CBSD location update	Pass
6.3.4.2.1	Unsuccessful Grant responseCode=400 (INTERFERENCE)	Pass
6.3.4.2.2	Unsuccessful Grant responseCode=401 (GRANT_CONFLICT)	Pass
6.4.4.1.1	Heartbeat Success Case (first Heartbeat Response)	Pass
6.4.4.2.1	Heartbeat responseCode=105 (DEREGISTER)	Pass
6.4.4.2.2	Heartbeat responseCode=500 (TERMINATED_GRANT)	Pass
6.4.4.2.3	Heartbeat responseCode=501 (SUSPENDED_GRANT) in First Heartbeat Response	Pass
6.4.4.2.4	Heartbeat responseCode=501 (SUSPENDED_GRANT) in Subsequent Heartbeat Response	Pass
6.4.4.2.5	Heartbeat responseCode=502 (UNSYNC_OP_PARAM)	Pass
6.4.4.3.1	Heartbeat Response Absent (First Heartbeat)	Pass
6.4.4.3.2	Heartbeat Response Absent (Subsequent Heartbeat)	Pass
6.4.4.4.1	Successful Grant Renewal in Heartbeat Test Case	Pass
6.5.4.2.1	Registration Response contains measReportConfig	Pass
6.5.4.2.3	Grant Response contains measReportConfig	Pass
6.5.4.2.4	Heartbeat Response contains measReportConfig	Pass
6.6.4.1.1	Successful Relinquishment	Pass
6.6.4.2.1	Unsuccessful Relinquishment, responseCode=102	Pass
6.6.4.3.1	Unsuccessful Relinquishment, responseCode=103	Pass
6.7.4.1.1	Successful Deregistration	Pass
6.7.4.2.1	Deregistration responseCode=102	Pass
6.7.4.3.1	Deregistration responseCode=103	Pass
6.8.4.1.1	Successful TLS connection between UUT and SAS Test Harness	Pass
6.8.4.2.1	TLS failure due to revoked certificate	Pass
6.8.4.2.2	TLS failure due to expired server certificate	Pass
6.8.4.2.3	TLS failure when SAS Test Harness certificate is issue by unknown CA	Pass
6.8.4.2.4	TLS failure when certificate at the SAS Test Harness is corrupted	Pass
7.1.4.1.1	UUT RF Transmit Power Measurement	Pass

Notes: none

Section 3. Equipment under test (EUT) details

3.1 Sample information

Receipt date	May 30, 2019
Nemko sample ID number	1, 2 and 3

3.2 EUT information

Product name	TEKO XRAN RadioUnit
CPE RF card model	XR19AX35WM2/48Y
Base Station model	THWPC-R-XT2AC
Serial numbers	1012482003
Revision number	1.0
Software version (SAS interface)	V1
Harness software version	1.0.0.3
Frequency band	CBRS band: 3550–3700 MHz
Type of modulation	QPSK½ to 64QAM
Power requirements	48 V_{DC} via PoE powered from 120 V_{AC} / 60 Hz

3.3 Product description and theory of operation

TEKO™ CellHub Distributed Radio System

- IT-centric offering for enterprise, in-building, venue and outdoor densification mobile connectivity
- Multiple operators/spectrum, including CBRS in single device
- Directly connects to a standard server locally or data center up to 12 miles away
- Domain proxy support to CBRS SAS

The TEKO CellHub is a JMA Wireless radio unit that supports high capacity, multi-channel CBRS (3550-3700 MHz, FCC Part 96) bands with, or without, simultaneous licensed carrier cellular bands. CellHubs support LTE for CBRS and licensed bands. As part of the JMA Wireless XRAN system it has the option of working in conjunction with new or existing distributed antenna systems.

Each CellHub acts as a CBSD under a Domain Proxy with compliance to the WInnForum SAS-CBSD interface, CBRS Alliance OnGo, and FCC Part 96 requirements. The Domain Proxy is provided as an independent software service module on the XRAN system. Multiple CellHubs can daisy chain miles apart to save on cabling and number of headend connections and homeruns.

3.4 EUT exercise details

SAS Installed and connected to Domain Proxy that acts on behalf of CBSDs. DNS, HTTP server on the same physical SAS's machine to perform SCS class tests. Spectrum analyzer connected to CBSDs' RF output.

3.5 EUT setup diagram

Basic system diagram shown below. XRAN supports multiple EPC connections for simultaneous Private LTE for CBRS and licensed carrier bands.

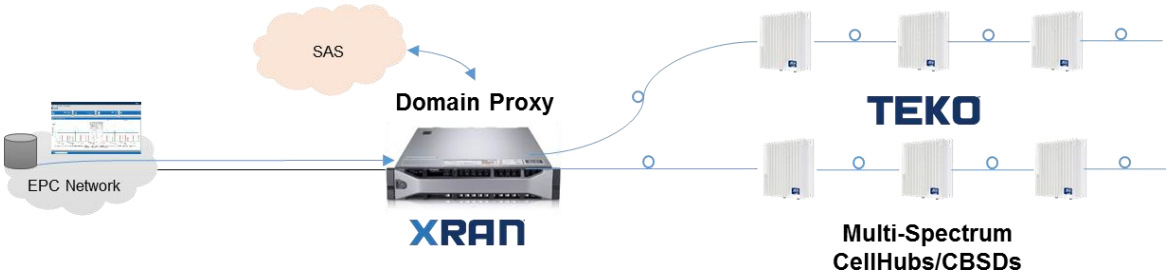

Figure 3.5-1: *Setup diagram*

3.6 EUT security per CBRS requirements

Requirement	Compliance
What communication protocol is used between the SAS and the CBSD?	The SAS-CBSD protocol is based on the HTTPS (HTTP over TLS version 1.2). The HTTPS protocol provides transport level assurance that a message has been received by the intended recipient. Communication includes mutual authentication using pki certificates.
How are communications initiated?	Per standard specification, SAS server discovery: SAS server URL is provided to CBSD's. CBSD via domain proxy communicate to server per URL provided and TLS mutual authentication will be performed. The CBSD/Domain Proxy initiating the TLS connection shall authenticate the SAS, and the SAS shall authenticate the CBSD/Domain Proxy.
How does the CBSD validate messages from the SAS?	Each massage session is encrypted and validated with TLSv1.2 and CA certificates verification. Messages also checked against protocol structure json.
How does the device handle failure to communicate or authenticate the SAS?	On communication failure/authentication, devices we re-try to communicate if fails, alarm will raise, and TX will stop.
How does the SAS validate messages from a CBSD?	Each massage session is encrypted and validated with TLSv1.2 and CA certificates verification. Messages also checked against protocol structure json.
What encryption method is used?	TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
How does the SAS ensure secure registration of protected devices?	By using user name and ID, also CPI signature can be usded.

Note: Protocols in accordance with: Document WINNF-TS-0016 Version V1.2.3 from October 31st, 2018

Section 4. Engineering considerations

4.1 Modifications incorporated in the EUT

There were no modifications performed to the EUT during this assessment.

4.2 Technical judgment

None

4.3 Deviations from laboratory tests procedures

No deviations were made from laboratory procedures.

Section 5. Test conditions

5.1 Atmospheric conditions

Temperature	15–30 °C
Relative humidity	20–75 %
Air pressure	860–1060 mbar

When it is impracticable to carry out tests under these conditions, a note to this effect stating the ambient temperature and relative humidity during the tests shall be recorded and stated.

5.2 Power supply range

The normal test voltage for equipment to be connected to the mains shall be the nominal mains voltage. For the purpose of the present document, the nominal voltage shall be the declared voltage, or any of the declared voltages ±5 %, for which the equipment was designed.

Section 6. Measurement uncertainty

6.1 Uncertainty of measurement

UKAS Lab 34 and TIA-603-B have been used as guidance for measurement uncertainty reasonable estimations with regards to previous experience and validation of data. Nemko Canada, Inc. follows these test methods in order to satisfy ISO/IEC 17025 requirements for estimation of uncertainty of measurement for wireless products.

Measurement uncertainty budgets for the tests are detailed below. Measurement uncertainty calculations assume a coverage factor of K = 2 with 95% certainty.

Table 6.1-1: *Measurement uncertainty*

Test name	Measurement uncertainty, dB
All antenna port measurements	0.55

Section 7. Test equipment

7.1 Test equipment list

Table 7.1-1: *Equipment list*

Equipment	Manufacturer	Model no.	Asset no.	Cal cycle	Next cal.
Spectrum analyzer	Rohde & Schwarz	FSU	FA001877	1 year	October 26, 2019

Section 8. Testing data

8.1 6.1.4.1.1 [WINNF.FT.C.REG.1] Multi-Step registration

8.1.1 Definitions and limits

6.1 CBSD Registration Process
This section provides test steps, conditions and procedures to test the conformance of the CBSD implementation for the CBSD Registration Procedure. A precondition is the CBSD has successfully discovered the SAS it wants to register with.
This test is mandatory for CBSD which supports multi-step registration. This test validates that each of the required parameters appear within the registration request message.

8.1.2 Test date

Start date	May 22, 2019

8.1.3 Observations, settings and special notes

None

8.1.4 Test data

Table 8.1-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT is in the Unregistered state	–	–
2	CBSD sends correct Registration request information, as specified in [n.5], to the SAS Test Harness: • The required userId, fccId and cbsdSerialNumber registration parameters shall be sent from the CBSD and conform to proper format and acceptable ranges. • Any REG-conditional or optional registration parameters that may be included in the message shall be verified that they conform to proper format and are within acceptable ranges. Note: It is outside the scope of this document to test the Registration information that is supplied via another means.	☒	☐
3	• SAS Test Harness sends a CBSD Registration Response as follows: o cbsdId = C o measReportConfig shall not be included o responseCode = 0	–	–
4	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–
5	Monitor the RF output of each UUT from start of test until 60 seconds after Step 3 is complete. This is the end of the test. Verify: • UUT shall not transmit RF	☒	☐

For the test log please refer to Section 9 of this test report.

8.2 6.1.4.1.3 [WINNF.FT.C.REG.3] Single-Step registration for Cat A CBSD

8.2.1 Definitions and limits

6.1 CBSD Registration Process
This section provides test steps, conditions and procedures to test the conformance of the CBSD implementation for the CBSD Registration Procedure. A precondition is the CBSD has successfully discovered the SAS it wants to register with.
This test is mandatory for CBSD which reports all Required and REG-Conditional parameters in the Registration request to the SAS, without CPI signed data.
This test validates that each of the required and REG-Conditional parameters appear within the registration request message.
For a Category A CBSD which determine its own location, the test lab and vendor must agree on the required evidence showing the UUT meets the location requirement. In lieu of location verification, the vendor shall supply their test approach/procedure along with compliance data.

8.2.2 Test date

Start date	May 22, 2019

8.2.3 Observations, settings and special notes

None

8.2.4 Test data

Table 8.2-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT is in the Unregistered state	–	–
2	CBSD sends Registration request to SAS Test Harness: all required and REG-Conditional parameter included (userId, fccId, cbsdSerialNumber, cbsdCategory, airInterface, installationParam, measCapability) for a Category A CBSD. • The required userId, fccId and cbsdSerialNumber and REG-Conditional cbsdCategory, airInterface, installationParam, and measCapability registration parameters shall be sent from the CBSD and conform to proper format and acceptable ranges. • Any optional registration parameters that may be included in the message shall be verified that they conform to proper format and are within acceptable ranges.	☒	☐
3	• SAS Test Harness sends a CBSD Registration Response as follows: o cbsdId = C o measReportConfig shall not be included o responseCode = 0	–	–
4	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–
5	Monitor the RF output of each UUT from start of test until 60 seconds after Step 3 is complete. This is the end of the test. Verify: • UUT shall not transmit RF	☒	☐

For the test log please refer to Section 9 of this test report.

8.3 6.1.4.1.7 [WINNF.FT.C.REG.7] Registration due to change of an installation parameter

8.3.1 Definitions and limits

6.1 CBSD Registration Process
This section provides test steps, conditions and procedures to test the conformance of the CBSD implementation for the CBSD Registration Procedure. A precondition is the CBSD has successfully discovered the SAS it wants to register with.
The purpose of this test is to verify the CBSD sends notification to the SAS when an installation parameter has been changed.
This test is limited to CBSDs that support a registration parameter change/update to be made at the CBSD.
Further, this test only applies to CBSD devices that allow a registration parameter change to be made prior to sending a deregistration.
This test is not valid for CBSDs requiring a deregistration prior to allowing a parameter change to be made (for example, (i) bring CBSD out of service (deregister), (ii) change registration parameter, (iii) bring CBSD back into service (register)). Refer to the deregistration test case [WINNF.FT.C.DRG.1].
This test is also not valid for CBSDs which require registration parameter updates to be made directly into the SAS via a SAS interface.

8.3.2 Test date

Start date	May 22, 2019

8.3.3 Observations, settings and special notes

None

8.3.4 Test data

Table 8.3-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness	–	–
2	UUT has successfully registered with SAS Test Harness	–	–
3	Change an installation parameters at the UUT (time T) Tester needs to record the current time at which the parameter change is executed.	–	–
4	Monitor the SAS-CBSD interface. UUT sends a deregistrationRequest to the SAS Test Harness The deregistration request shall be sent within (T + 60 seconds) from step 3.	☒	☐

For the test log please refer to Section 9 of this test report.

8.4 6.1.4.2.1 [WINNF.FT.C.REG.8] Missing Required parameters (responseCode 102)

8.4.1 Definitions and limits

6.1 CBSD Registration Process
CBSD under test cannot be expected to generate a message with a missing or invalid parameter. To test for responseCode not equal to 0, the SAS Test Harness will respond to a valid registrationRequest message with a registrationResponse with a non-zero responseCode.
The purpose of these tests is to ensure that the CBSD does not transmit when a responseCode other than 0 is received. The information sent in the registration request message is not important, only that it shall conform to the protocol.
Missing/Invalid response codes are tested by injecting those responseCodes into the SAS Test Harness generated response message, even though the UUT has sent a valid message
The following are the test execution steps where the Registration response contains responseCode (R) = 102 .

8.4.2 Test date

Start date	May 22, 2019

8.4.3 Observations, settings and special notes

None

8.4.4 Test data

Table 8.4-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT is in the Unregistered state	–	–
2	CBSD sends a Registration request to SAS Test Harness.	–	–
3	SAS Test Harness rejects the request by sending a CBSD Registration Response as follows: ○ SAS response does not include a cbsdId. ○ responseCode = R	–	–
4	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–
5	Monitor the RF output of each UUT from start of test until 60 seconds after Step 3 is complete. This is the end of the test. Verify: • UUT shall not transmit RF	☒	☐

For the test log please refer to Section 9 of this test report.

8.5 6.1.4.2.3 [WINNF.FT.C.REG.10] Pending registration (responseCode 200)

8.5.1 Definitions and limits

6.1 CBSD Registration Process
CBSD under test cannot be expected to generate a message with a missing or invalid parameter. To test for responseCode not equal to 0, the SAS Test Harness will respond to a valid registrationRequest message with a registrationResponse with a non-zero responseCode.
The purpose of these tests is to ensure that the CBSD does not transmit when a responseCode other than 0 is received. The information sent in the registration request message is not important, only that it shall conform to the protocol.
Missing/Invalid response codes are tested by injecting those responseCodes into the SAS Test Harness generated response message, even though the UUT has sent a valid message
The same steps provided for WINNF.FT.C.REG.8 shall be executed for this test, with the exception that the Registration response contains responseCode (R) = 200.

8.5.2 Test date

Start date	May 22, 2019

8.5.3 Observations, settings and special notes

None

8.5.4 Test data

Table 8.5-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT is in the Unregistered state	–	–
2	CBSD sends a Registration request to SAS Test Harness.	–	–
3	SAS Test Harness rejects the request by sending a CBSD Registration Response as follows: o SAS response does not include a cbsdId. o responseCode = 200	–	–
4	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–
5	Monitor the RF output of each UUT from start of test until 60 seconds after Step 3 is complete. This is the end of the test. Verify: • UUT shall not transmit RF	☒	☐

For the test log please refer to Section 9 of this test report.

8.6 6.1.4.2.5 [WINNF.FT.C.REG.12] Invalid parameters (responseCode 103)

8.6.1 Definitions and limits

6.1 CBSD Registration Process
CBSD under test cannot be expected to generate a message with a missing or invalid parameter. To test for responseCode not equal to 0, the SAS Test Harness will respond to a valid registrationRequest message with a registrationResponse with a non-zero responseCode.
The purpose of these tests is to ensure that the CBSD does not transmit when a responseCode other than 0 is received. The information sent in the registration request message is not important, only that it shall conform to the protocol.
Missing/Invalid response codes are tested by injecting those responseCodes into the SAS Test Harness generated response message, even though the UUT has sent a valid message
The same steps provided for WINNF.FT.C.REG.8 shall be executed for this test, with the exception that the Registration response contains responseCode (R) = 103.

8.6.2 Test date

Start date	May 22, 2019

8.6.3 Observations, settings and special notes

None

8.6.4 Test data

Table 8.6-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT is in the Unregistered state	–	–
2	CBSD sends a Registration request to SAS Test Harness.	–	–
3	SAS Test Harness rejects the request by sending a CBSD Registration Response as follows: o SAS response does not include a cbsdId. o responseCode = 103	–	–
4	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–
5	Monitor the RF output of each UUT from start of test until 60 seconds after Step 3 is complete. This is the end of the test. Verify: • UUT shall not transmit RF	☒	☐

For the test log please refer to Section 9 of this test report.

8.7 6.1.4.2.7 [WINNF.FT.C.REG.14] Blacklisted CBSD (responseCode 101)

8.7.1 Definitions and limits

6.1 CBSD Registration Process
CBSD under test cannot be expected to generate a message with a missing or invalid parameter. To test for responseCode not equal to 0, the SAS Test Harness will respond to a valid registrationRequest message with a registrationResponse with a non-zero responseCode.
The purpose of these tests is to ensure that the CBSD does not transmit when a responseCode other than 0 is received. The information sent in the registration request message is not important, only that it shall conform to the protocol.
Missing/Invalid response codes are tested by injecting those responseCodes into the SAS Test Harness generated response message, even though the UUT has sent a valid message
The same steps provided for WINNF.FT.D.REG.9 shall be executed for this test, with the exception that the Registration response contains responseCode R1 = 0 for CBSD1 and R2 = 101 for CBSD2.

8.7.2 Test date

Start date	May 22, 2019

8.7.3 Observations, settings and special notes

None

8.7.4 Test data

Table 8.7-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT is in the Unregistered state	–	–
2	CBSD sends a Registration request to SAS Test Harness.	–	–
3	SAS Test Harness rejects the request by sending a CBSD Registration Response as follows: o SAS response does not include a cbsdId. o responseCode = 101	–	–
4	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–
5	Monitor the RF output of each UUT from start of test until 60 seconds after Step 3 is complete. This is the end of the test. Verify: • UUT shall not transmit RF	☒	☐

For the test log please refer to Section 9 of this test report.

8.8 6.1.4.2.9 [WINNF.FT.C.REG.16] Unsupported SAS protocol version (responseCode 100)

8.8.1 Definitions and limits

6.1 CBSD Registration Process
CBSD under test cannot be expected to generate a message with a missing or invalid parameter. To test for responseCode not equal to 0, the SAS Test Harness will respond to a valid registrationRequest message with a registrationResponse with a non-zero responseCode.
The purpose of these tests is to ensure that the CBSD does not transmit when a responseCode other than 0 is received. The information sent in the registration request message is not important, only that it shall conform to the protocol.
Missing/Invalid response codes are tested by injecting those responseCodes into the SAS Test Harness generated response message, even though the UUT has sent a valid message
The same steps provided for WINNF.FT.D.REG.9 shall be executed for this test, with the exception that the Registration response contains responseCode (Ri) = 100 for each CBSD.

8.8.2 Test date

Start date	October 1, 2018

8.8.3 Observations, settings and special notes

None

8.8.4 Test data

Table 8.8-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT is in the Unregistered state	–	–
2	CBSD sends a Registration request to SAS Test Harness.	–	–
3	SAS Test Harness rejects the request by sending a CBSD Registration Response as follows: o SAS response does not include a cbsdId. o responseCode = 100	–	–
4	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–
5	Monitor the RF output of each UUT from start of test until 60 seconds after Step 3 is complete. This is the end of the test. Verify: • UUT shall not transmit RF	☒	☐

For the test log please refer to Section 9 of this test report.

8.9 6.1.4.2.11 [WINNF.FT.C.REG.18] Group Error (responseCode 201)

8.9.1 Definitions and limits

6.1 CBSD Registration Process
CBSD under test cannot be expected to generate a message with a missing or invalid parameter. To test for responseCode not equal to 0, the SAS Test Harness will respond to a valid registrationRequest message with a registrationResponse with a non-zero responseCode.
The purpose of these tests is to ensure that the CBSD does not transmit when a responseCode other than 0 is received. The information sent in the registration request message is not important, only that it shall conform to the protocol.
Missing/Invalid response codes are tested by injecting those responseCodes into the SAS Test Harness generated response message, even though the UUT has sent a valid message
The registrationRequest groupingParam is an optional field and will be validated by the SAS Test Harness if provided in the Registration Request message.
This test will validate that the CBSD will remain Unregistered after receiving responseCode 201.
The same steps provided for WINNF.FT.C.REG.8 shall be executed for this test, with the exception that the Registration response contains responseCode (R) = 201.

8.9.2 Test date

Start date	May 22, 2019

8.9.3 Observations, settings and special notes

None

8.9.4 Test data

Table 8.9-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT is in the Unregistered state	–	–
2	CBSD sends a Registration request to SAS Test Harness.	–	–
3	SAS Test Harness rejects the request by sending a CBSD Registration Response as follows: o SAS response does not include a cbsdId. o responseCode = 201	–	–
4	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–
5	Monitor the RF output of each UUT from start of test until 60 seconds after Step 3 is complete. This is the end of the test. Verify: • UUT shall not transmit RF	☒	☐

For the test log please refer to Section 9 of this test report.

8.10 6.1.4.3.1 [WINNF.FT.C.REG.20] Category A CBSD location update

8.10.1 Definitions and limits

6.1 CBSD Registration Process
This section is specific to Category A CBSDs that do not require professional installation. The requirement is for the Category A (non-professionally installed) to report to the SAS any location change exceeding a distance of 50m horizontally or 3m vertically within a 60 second window. It is left to the CBSD vendor and certification lab to generate the required evidence showing the UUT meets the requirement.
The test case ID is provided as a means to ensure that evidence is provided showing compliance to this requirement.

8.10.2 Test date

Start date	May 22, 2019

8.10.3 Observations, settings and special notes

None

8.10.4 Test data

Table 8.10-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness	–	–
2	UUT has successfully registered with SAS Test Harness	–	–
3	Change an installation parameter at the UUT (time T) ○ Tester needs to record the current time at which the parameter change is executed.	–	–
4	Monitor the SAS-CBSD interface. UUT sends a deregistrationRequest to the SAS Test Harness The deregistration request shall be sent within (T + 60 seconds) from step 3.	☒	☐

For the test log please refer to Section 9 of this test report.

8.11 6.3.4.2.1 [WINNF.FT.C.GRA.1] Unsuccessful Grant responseCode=400 (INTERFERENCE)

8.11.1 Definitions and limits

6.3 CBSD Spectrum Grant Process
This section provides test steps, condition and procedures to test the conformance of the CBSD implementation for the CBSD Spectrum Grant Procedure. A precondition is the CBSD has successfully discovered the SAS it wants to communicate with.
Each test generates a CBSD spectrum grant request and validates the CBSD takes the appropriate action following the SAS spectrum grant response.
The test cases in this section are for verifying the handling of CBSD for various responseCodes in response from the-SAS Test Harness.
The actions taken in response of any responseCode are beyond the scope of this document unless mentioned in the test procedure.

8.11.2 Test date

Start date	May 22, 2019

8.11.3 Observations, settings and special notes

None

8.11.4 Test data

Table 8.11-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has registered successfully with SAS Test Harness, with cbsdId = C	–	–
2	UUT sends valid Grant Request.	–	–
3	SAS Test Harness sends a Grant Response message, including • cbsdId=C • responseCode = R	–	–
4	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–
5	Monitor the RF output of the UUT from start of test until 60 seconds after Step 3 is complete. This is the end of the test. Verify: • UUT shall not transmit RF	☒	☐

For the test log please refer to Section 9 of this test report.

8.12 6.3.4.2.2 [WINNF.FT.C.GRA.2] Unsuccessful Grant responseCode=401 (GRANT_CONFLICT)

8.12.1 Definitions and limits

6.3 CBSD Spectrum Grant Process
This section provides test steps, condition and procedures to test the conformance of the CBSD implementation for the CBSD Spectrum Grant Procedure. A precondition is the CBSD has successfully discovered the SAS it wants to communicate with.
Each test generates a CBSD spectrum grant request and validates the CBSD takes the appropriate action following the SAS spectrum grant response.
The test cases in this section are for verifying the handling of CBSD for various responseCodes in response from the-SAS Test Harness.
The actions taken in response of any responseCode are beyond the scope of this document unless mentioned in the test procedure.
The same steps provided for WINNF.FT.C.GRA.1 shall be executed for this test, with the exception that the Grant response contains responseCode (R) = 401.

8.12.2 Test date

Start date	May 22, 2019

8.12.3 Observations, settings and special notes

None

8.12.4 Test data

Table 8.12-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has registered successfully with SAS Test Harness, with cbsdId = C	–	–
2	UUT sends valid Grant Request.	–	–
3	SAS Test Harness sends a Grant Response message, including • cbsdId=C • responseCode (R) = 401	–	–
4	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–
5	Monitor the RF output of the UUT from start of test until 60 seconds after Step 3 is complete. This is the end of the test. Verify: • UUT shall not transmit RF	☒	☐

For the test log please refer to Section 9 of this test report.

8.13 6.4.4.1.1 [WINNF.FT.C.HBT.1] Heartbeat Success Case (first Heartbeat Response)

8.13.1 Definitions and limits

6.4 CBSD Heart Beat Process
This section provides procedures for testing CBSD behavior during the Heartbeat Process. It assumes as precondition that CBSD has successfully discovered the SAS that it wants to register with, has successfully registered, has a successful Grant request, and is in the Granted or Authorized state.
The test cases in this section test the success path for the Heartbeat process. The SAS Test Harness shall use a heartBeatInterval of 60 seconds, unless specifically provided in the test case.
This test case incorporates validation of successful Spectrum Inquiry messaging (if present) and successful Grant messaging into the Heartbeat Success case.

8.13.2 Test date

Start date	May 22, 2019

8.13.3 Observations, settings and special notes

None

8.13.4 Test data

Table 8.13-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has registered successfully with SAS Test Harness, with cbsdId = C	–	–
2	UUT sends a message: • If message is type Spectrum Inquiry Request, go to step 3, or • If message is type Grant Request, go to step 5	–	–
3	UUT sends Spectrum Inquiry Request. Validate: • cbsdId = C • List of frequencyRange objects sent by UUT are within the CBRS frequency range	☒	☐
4	SAS Test Harness sends a Spectrum Inquiry Response message, including the following parameters: • cbsdId = C • availableChannel is an array of availableChannel objects • responseCode = 0	–	–
5	UUT sends Grant Request message. Validate: • cbsdId = C • maxEIRP is at or below the limit appropriate for CBSD category as defined by Part 96 • operationFrequencyRange, F, sent by UUT is a valid range within the CBRS band	☒	☐
6	SAS Test Harness sends a Grant Response message, including the parameters: • cbsdId = C • grantId = G = a valid grant ID • grantExpireTime = UTC time greater than duration of the test • responseCode = 0	–	–
7	UUT sends a first Heartbeat Request message. Verify Heartbeat Request message is formatted correctly, including: • cbsdId = C • grantId = G • operationState = “GRANTED”	☒	☐

Step	Test Execution Steps	Pass	Fail
8	SAS Test Harness sends a Heartbeat Response message, with the following parameters: • cbsdId = C • grantId = G • transmitExpireTime = current UTC time + 200 seconds • responseCode = 0	–	–
9	For further Heartbeat Request messages sent from UUT after completion of step 8, validate message is sent within latest specified heartbeatInterval, and: • cbsdId = C • grantId = G • operationState = "AUTHORIZED" and SAS Test Harness responds with a Heartbeat Response message including the following parameters: • cbsdId = C • grantId = G • transmitExpireTime = current UTC time + 200 seconds • responseCode = 0	☒	☐
10	Monitor the RF output of the UUT from start of test until UUT transmission commences. Verify: • UUT does not transmit at any time prior to completion of the first heartbeat response • UUT transmits after step 8 is complete, and its transmission is limited to within the bandwidth range F.	☒	☐

For the test log please refer to Section 9 of this test report.

8.14 6.4.4.2.1 [WINNF.FT.C.HBT.3] Heartbeat responseCode=105 (DEREGISTER)

8.14.1 Definitions and limits

6.4 CBSD Heart Beat Process
This section provides procedures for testing CBSD behavior during the Heartbeat Process. It assumes as precondition that CBSD has successfully discovered the SAS that it wants to register with, has successfully registered, has a successful Grant request, and is in the Granted or Authorized state.
The test cases in this section cover Heartbeat Response messages with non-zero responseCodes. Part of the pass/fail criteria of these test cases is the cessation of all UUT RF transmission. Therefore, in all test cases, after the non-zero responseCode is sent, the SAS Test Harness shall not allow any new Grant Request from the UUT to succeed.

8.14.2 Test date

Start date	May 22, 2019

8.14.3 Observations, settings and special notes

None

8.14.4 Test data

Table 8.14-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has registered successfully with SAS Test Harness • UUT has a valid single grant as follows: o valid cbsdId = C o valid grantId = G o grant is for frequency range F, power P o grantExpireTime = UTC time greater than duration of the test • UUT is in AUTHORIZED state and is transmitting within the grant bandwidth F on RF interface	–	–
2	UUT sends a Heartbeat Request message. Ensure Heartbeat Request message is sent within Heartbeat Interval specified in the latest Heartbeat Response, and formatted correctly, including: • cbsdId = C • grantId = G • operationState = “AUTHORIZED”	☒	☐
3	SAS Test Harness sends a Heartbeat Response message, including the following parameters: • cbsdId = C • grantId = G • transmitExpireTime = T = Current UTC time • responseCode = 105 (DEREGISTER)	–	–
4	After completion of step 3, SAS Test Harness shall not allow any further grants to the UUT.	–	–
5	Monitor the RF output of the UUT. Verify: • UUT shall stop transmission within (T + 60 seconds) of completion of step 3	☒	☐

For the test log please refer to Section 9 of this test report.

8.15 6.4.4.2.2 [WINNF.FT.C.HBT.4] Heartbeat responseCode=500 (TERMINATED_GRANT)

8.15.1 Definitions and limits

6.4 CBSD Heart Beat Process
This section provides procedures for testing CBSD behavior during the Heartbeat Process. It assumes as precondition that CBSD has successfully discovered the SAS that it wants to register with, has successfully registered, has a successful Grant request, and is in the Granted or Authorized state.
The test cases in this section cover Heartbeat Response messages with non-zero responseCodes. Part of the pass/fail criteria of these test cases is the cessation of all UUT RF transmission. Therefore, in all test cases, after the non-zero responseCode is sent, the SAS Test Harness shall not allow any new Grant Request from the UUT to succeed.

8.15.2 Test date

Start date	May 22, 2019

8.15.3 Observations, settings and special notes

None

8.15.4 Test data

Table 8.15-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has registered successfully with SAS Test Harness • UUT has a valid single grant as follows: o valid cbsdId = C o valid grantId = G o grant is for frequency range F, power P o grantExpireTime = UTC time greater than duration of the test • UUT is in AUTHORIZED state and is transmitting within the grant bandwidth F on RF interface	–	–
2	UUT sends a Heartbeat Request message. Ensure Heartbeat Request message is sent within latest specified heartbeatInterval, and is formatted correctly, including: • cbsdId = C • grantId = G • operationState = “AUTHORIZED”	☒	☐
3	SAS Test Harness sends a Heartbeat Response message, including the following parameters: • cbsdId = C • grantId = G • transmitExpireTime = T = current UTC time • responseCode = 500 (TERMINATED_GRANT)	–	–
4	After completion of step 3, SAS Test Harness shall not allow any further grants to the UUT.	–	–
5	Monitor the RF output of the UUT. Verify: • UUT shall stop transmission within (T + 60 seconds) of completion of step 3	☒	☐

For the test log please refer to Section 9 of this test report.

8.16 6.4.4.2.3 [WINNF.FT.C.HBT.5] Heartbeat responseCode=501 (SUSPENDED_GRANT) in First Heartbeat Response

8.16.1 Definitions and limits

6.4 CBSD Heart Beat Process
This section provides procedures for testing CBSD behavior during the Heartbeat Process. It assumes as precondition that CBSD has successfully discovered the SAS that it wants to register with, has successfully registered, has a successful Grant request, and is in the Granted or Authorized state.
The test cases in this section cover Heartbeat Response messages with non-zero responseCodes. Part of the pass/fail criteria of these test cases is the cessation of all UUT RF transmission. Therefore, in all test cases, after the non-zero responseCode is sent, the SAS Test Harness shall not allow any new Grant Request from the UUT to succeed.

8.16.2 Test date

Start date	May 22, 2019

8.16.3 Observations, settings and special notes

None

8.16.4 Test data

Table 8.16-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has registered successfully with SAS Test Harness • UUT has a valid single grant as follows: o valid cbsdId = C o valid grantId = G o grant is for frequency range F, power P o grantExpireTime = UTC time greater than duration of the test • UUT is in GRANTED, but not AUTHORIZED state (i.e. has not performed its first Heartbeat Request)	–	–
2	UUT sends a Heartbeat Request message. Verify Heartbeat Request message is formatted correctly, including: • cbsdId = C • grantId = G • operationState = “GRANTED”	☒	☐
3	SAS Test Harness sends a Heartbeat Response message, including the parameters: • cbsdId = C • grantId = G • transmitExpireTime = T = current UTC time • responseCode = 501 (SUSPENDED_GRANT)	–	–
4	After completion of step 3, SAS Test Harness shall not allow any further grants to the UUT.	–	–

Step	Test Execution Steps	Pass	Fail
5	Monitor the SAS-CBSD interface. Verify either A OR B occurs: A. UUT sends a Heartbeat Request message. Ensure message is sent within latest specified heartbeatInterval, and is correctly formatted with parameters: • cbsdId = C • grantId = G • operationState = "GRANTED" B. UUT sends a Relinquishment request message. Ensure message is correctly formatted with parameters: • cbdsId = C • grantId = G Monitor the RF output of the UUT. Verify: • UUT does not transmit at any time	☒	☐

For the test log please refer to Section 9 of this test report.

8.17 6.4.4.2.4 [WINNF.FT.C.HBT.6] Heartbeat responseCode=501 (SUSPENDED_GRANT) in Subsequent Heartbeat Response

8.17.1 Definitions and limits

6.4 CBSD Heart Beat Process
This section provides procedures for testing CBSD behavior during the Heartbeat Process. It assumes as precondition that CBSD has successfully discovered the SAS that it wants to register with, has successfully registered, has a successful Grant request, and is in the Granted or Authorized state.
The test cases in this section cover Heartbeat Response messages with non-zero responseCodes. Part of the pass/fail criteria of these test cases is the cessation of all UUT RF transmission. Therefore, in all test cases, after the non-zero responseCode is sent, the SAS Test Harness shall not allow any new Grant Request from the UUT to succeed.

8.17.2 Test date

Start date	May 22, 2019

8.17.3 Observations, settings and special notes

None

8.17.4 Test data

Table 8.17-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has registered successfully with SAS Test Harness • UUT has a valid single grant as follows: o valid cbsdId = C o valid grantId = G o grant is for frequency range F, power P o grantExpireTime = UTC time greater than duration of the test • UUT is in GRANTED, but not AUTHORIZED state (i.e. has not performed its first Heartbeat Request)	–	–
2	UUT sends a Heartbeat Request message. Verify Heartbeat Request message is sent within latest specified heartbeatInterval, and is formatted correctly, including: • cbsdId = C • grantId = G • operationState = “AUTHORIZED”	☒	☐
3	SAS Test Harness sends a Heartbeat Response message, including the following parameters: • cbsdId = C • grantId = G • transmitExpireTime = T = current UTC time • responseCode = 501 (SUSPENDED_GRANT)	–	–
4	After completion of step 3, SAS Test Harness shall not allow any further grants to the UUT.	–	–

Step	Test Execution Steps	Pass	Fail
5	Monitor the SAS-CBSD interface. Verify either A OR B occurs: A. UUT sends a Heartbeat Request message. Ensure message is sent within latest specified heartbeatInterval, and is correctly formatted with parameters: • cbsdId = C • grantId = G • operationState = "GRANTED" B. UUT sends a Relinquishment Request message. Ensure message is correctly formatted with parameters: • cbdsId = C • grantId = G Monitor the RF output of the UUT. Verify: • UUT shall stop transmission within (T + 60 seconds) of completion of step 3	☒	☐

For the test log please refer to Section 9 of this test report.

8.18 6.4.4.2.5 [WINNF.FT.C.HBT.7] Heartbeat responseCode=502 (UNSYNC_OP_PARAM)

8.18.1 Definitions and limits

6.4 CBSD Heart Beat Process
This section provides procedures for testing CBSD behavior during the Heartbeat Process. It assumes as precondition that CBSD has successfully discovered the SAS that it wants to register with, has successfully registered, has a successful Grant request, and is in the Granted or Authorized state.
The test cases in this section cover Heartbeat Response messages with non-zero responseCodes. Part of the pass/fail criteria of these test cases is the cessation of all UUT RF transmission. Therefore, in all test cases, after the non-zero responseCode is sent, the SAS Test Harness shall not allow any new Grant Request from the UUT to succeed.

8.18.2 Test date

Start date	May 22, 2019

8.18.3 Observations, settings and special notes

None

8.18.4 Test data

Table 8.18-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has registered successfully with SAS Test Harness • UUT has a valid single grant as follows: o valid cbsdId = C o valid grantId = G o grant is for frequency range F, power P o grantExpireTime = UTC time greater than duration of the test • UUT is in AUTHORIZED state and is transmitting within the grant bandwidth F on RF interface	–	–
2	UUT sends a Heartbeat Request message. Verify Heartbeat Request message is sent within latest specified heartbeatInterval, and is formatted correctly, including: • cbsdId = C • grantId = G • operationState = “AUTHORIZED”	☒	☐
3	SAS Test Harness sends a Heartbeat Response message, including the following parameters: • cbsdId = C • grantId = G • transmitExpireTime = T = Current UTC Time • responseCode = 502 (UNSYNC_OP_PARAM)	–	–
4	After completion of step 3, SAS Test Harness shall not allow any further grants to the UUT.	–	–
5	Monitor the SAS-CBSD interface. Verify: • UUT sends a Grant Relinquishment Request message. Verify message is correctly formatted with parameters: o cbdsId = C o grantId = G Monitor the RF output of the UUT. Verify: • UUT shall stop transmission within (T+60) seconds of completion of step 3.	☒	☐

For the test log please refer to Section 9 of this test report.

8.19 6.4.4.3.1 [WINNF.FT.C.HBT.9] Heartbeat Response Absent (First Heartbeat)

8.19.1 Definitions and limits

6.4 CBSD Heart Beat Process
This section provides procedures for testing CBSD behavior during the Heartbeat Process. It assumes as precondition that CBSD has successfully discovered the SAS that it wants to register with, has successfully registered, has a successful Grant request, and is in the Granted or Authorized state.
These test cases cover the case where communication is lost between the UUT and the SAS during the Heartbeat Process.

8.19.2 Test date

Start date	May 22, 2019

8.19.3 Observations, settings and special notes

None

8.19.4 Test data

Table 8.19-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has registered successfully with SAS Test Harness • UUT has a valid single grant as follows: o valid cbsdId = C o valid grantId = G o grant is for frequency range F, power P o grantExpireTime = UTC time greater than duration of the test • UUT is in GRANTED, but not AUTHORIZED state (i.e. has not performed its first Heartbeat Request)	–	–
2	UUT sends a Heartbeat Request message. Ensure Heartbeat Request message is sent within latest specified heartbeatInterval, and is formatted correctly, including: • cbsdId = C • grantId = G • operationState = “GRANTED”	☒	☐
3	After completion of Step 2, SAS Test Harness does not respond to any further messages from UUT to simulate loss of network connection	–	–
4	Monitor the RF output of the UUT from start of test to 60 seconds after step 3. Verify: • At any time during the test, UUT shall not transmit on RF interface	☒	☐

For the test log please refer to Section 9 of this test report.

8.20 6.4.4.3.2 [WINNF.FT.C.HBT.10] Heartbeat Response Absent (Subsequent Heartbeat)

8.20.1 Definitions and limits

6.4 CBSD Heart Beat Process
This section provides procedures for testing CBSD behavior during the Heartbeat Process. It assumes as precondition that CBSD has successfully discovered the SAS that it wants to register with, has successfully registered, has a successful Grant request, and is in the Granted or Authorized state.
These test cases cover the case where communication is lost between the UUT and the SAS during the Heartbeat Process.

8.20.2 Test date

Start date	May 22, 2019

8.20.3 Observations, settings and special notes

None

8.20.4 Test data

Table 8.20-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has registered successfully with SAS Test Harness • UUT has a valid single grant as follows: o valid cbsdId = C o valid grantId = G o grant is for frequency range F, power P o grantExpireTime = UTC time greater than duration of the test • UUT is in AUTHORIZED state and is transmitting within the grant bandwidth F on RF interface	–	–
2	UUT sends a Heartbeat Request message. Verify Heartbeat Request message issent within the latest specified heartbeatInterval, and is formatted correctly, including: • cbsdId = C • grantId = G • operationState = “AUTHORIZED”	☒	☐
3	SAS Test Harness sends a Heartbeat Response message, with the following parameters: • cbsdId = C • grantId = G • transmitExpireTime = current UTC time + 200 seconds • responseCode = 0	–	–
4	After completion of Step 3, SAS Test Harness does not respond to any further messages from UUT	–	–
5	Monitor the RF output of the UUT. Verify: • UUT shall stop all transmission on RF interface within (transmitExpireTime + 60 seconds), using the transmitExpireTime sent in Step 3.	☒	☐

For the test log please refer to Section 9 of this test report.

8.21 6.4.4.4.1 [WINNF.FT.C.HBT.11] Successful Grant Renewal in Heartbeat Test Case

8.21.1 Definitions and limits

6.4.4.4 Heartbeat Grant Renewal Cases
Test cases in this section test Grant Renewal within the Heartbeat Process.

8.21.2 Test date

Start date	May 22, 2019

8.21.3 Observations, settings and special notes

None

8.21.4 Test data

Table 8.21-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has registered successfully with SAS Test Harness • UUT has a valid single grant as follows: o valid cbsdId = C o valid grantId = G o grant is for frequency range F, power P • UUT is in AUTHORIZED state and is transmitting within the grant bandwidth F on RF interface. • Grant has the following parameters at the start of the test: o grantExpireTime =UTC time equal to time at start of test + 300 seconds = Tgrant_expire o transmitExpireTime = UTC time equal to time at start of test + 200 seconds o heartbeatInterval = 60 seconds	–	–
2	UUT sends a Heartbeat Request message. If Heartbeat Request message contains grantRenew = TRUE, go to Step 6, else go to Step 3.	–	–
3	Verify Heartbeat Request message is sent within the latest specified heartbeatInterval, and is formatted correctly, including: • cbsdId = C • grantId = G • operationState = “AUTHORIZED”	☒	☐
4	SAS Test Harness sends a Heartbeat Response message, with the following parameters: • cbsdId = C • grantId = G • transmitExpireTime = current UTC + 200 seconds • grantExpireTime = same as Step 1 • responseCode = 0	–	–
5	Go to Step 2	–	–
5	Verify Heartbeat Request message is sent within the latest specified heartbeatInterval, and is formatted correctly, including: • cbsdId = C • grantId = G	☒	☐

For the test log please refer to Section 9 of this test report.

8.22 6.5.4.2.1 [WINNF.FT.C.MES.1] Registration Response contains measReportConfig

8.22.1 Definitions and limits

6.5 CBSD Measurement Report
This section explains test steps/condition/procedure for CBSD behavior for Measurement Reports.
The main test cases for Measurement Report are outlined below, in terms of Measurement Report Stimulus (in a Response message from SAS) and a Measurement Report Response (in the subsequent Request message from the UUT).
Devices which support one measurement capability must satisfy the test cases mandatory for that measurement capability. Devices which support multiple measurement capabilities must satisfy the test cases mandatory for each type of supported measurement capability.
This test case is mandatory for CBSD supporting RECEIVED_POWER_WITHOUT_GRANT.

8.22.2 Test date

Start date	May 22, 2019

8.22.3 Observations, settings and special notes

None

8.22.4 Test data

Table 8.22-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness	–	–
2	UUT sends a Registration Request message. Validate the Registration Request message is formatted correctly, including: • userId is present and correct • fccId is present and correct • cbsdSerialNumber is present and correct • measCapability = “RECEIVED_POWER_WITHOUT_GRANT”	☒	☐
3	SAS Test Harness sends a Registration Response message, with the following parameters: • cbsdId = C = valid cbsdId for this UUT • measReportConfig= “RECEIVED_POWER_WITHOUT_GRANT” • responseCode = 0	–	–
4	UUT sends a message: • If message is type Spectrum Inquiry Request, go to step 5, or • If message is type Grant Request, go to step 7	–	–
5	UUT sends message type Spectrum Inquiry Request. Verify message contains all required parameters properly formatted, and specifically: • cbsdId = C • measReport is present, and is a properly formatted rcvdPowerMeasReport.	☒	☐
6	SAS Test Harness sends a Spectrum Inquiry Response, with the following parameters: • cbsdId = C • availableChannel is an array of availableChannel objects • responseCode = 0	–	–

Step	Test Execution Steps	Pass	Fail
7	UUT sends message type Grant Request message. Verify message contains all required parameters properly formatted, and specifically: • cbsdId = C • measReport is present, and is a properly formatted rcvdPowerMeasReport.	☒	☐

For the test log please refer to Section 9 of this test report.

8.23 6.5.4.2.3 [WINNF.FT.C.MES.3] Grant Response contains measReportConfig

8.23.1 Definitions and limits

6.5 CBSD Measurement Report
This section explains test steps/condition/procedure for CBSD behavior for Measurement Reports.
The main test cases for Measurement Report are outlined below, in terms of Measurement Report Stimulus (in a Response message from SAS) and a Measurement Report Response (in the subsequent Request message from the UUT).
Devices which support one measurement capability must satisfy the test cases mandatory for that measurement capability. Devices which support multiple measurement capabilities must satisfy the test cases mandatory for each type of supported measurement capability.
This test case is mandatory for UUT supporting RECEIVED_POWER_WITH_GRANT measurement reports.

8.23.2 Test date

Start date	May 22, 2019

8.23.3 Observations, settings and special notes

None

8.23.4 Test data

Table 8.23-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT has successfully registered with SAS Test Harness, with cbsdId=C and measCapability = "RECEIVED_POWER_WITH_GRANT"	–	–
2	UUT sends a Grant Request message. Verify Grant Request message contains all required parameters properly formatted, and specifically: • cbsdId = C • operationParam is present and format is valid	☒	☐
3	SAS Test Harness sends a Grant Response message, with the following parameters: • cbsdId = C • grantId = G = valid grant ID • grantExpireTime = UTC time in the future • heartbeatInterval = 60 seconds • measReportConfig= "RECEIVED_POWER_WITH_GRANT" • operationParam is set to valid operating parameters • channelType = "GAA" • responseCode = 0	–	–
4	UUT sends a Heartbeat Request message. Verify message contains all required parameters properly formatted, and specifically: • cbsdId = C • grantId = G • operationState = "GRANTED"	☒	☐
5	If Heartbeat Request message (step 4) contains measReport object, then: • verify measReport is properly formatted as object rcvdPowerMeasReport • end test, with PASS result else, if Heartbeat Request message (step 4) does not contain measReport object, then: If number of Heartbeat Requests sent by UUT after Step 3 is = 5, then stop test with result of FAIL	☒	☐

Step	Test Execution Steps	Pass	Fail
6	SAS Test Harness sends a Heartbeat Response message, containing all required parameters properly formatted, and specifically: • cbsdId = C • grantId = G • transmitExpireTime = current UTC time + 200 seconds • responseCode = 0 Go to Step 4, above	–	–

For the test log please refer to Section 9 of this test report.

8.24 6.5.4.2.4 [WINNF.FT.C.MES.4] Heartbeat Response contains measReportConfig

8.24.1 Definitions and limits

6.5 CBSD Measurement Report
This section explains test steps/condition/procedure for CBSD behavior for Measurement Reports.
The main test cases for Measurement Report are outlined below, in terms of Measurement Report Stimulus (in a Response message from SAS) and a Measurement Report Response (in the subsequent Request message from the UUT).
Devices which support one measurement capability must satisfy the test cases mandatory for that measurement capability. Devices which support multiple measurement capabilities must satisfy the test cases mandatory for each type of supported measurement capability.
This test case is mandatory for UUT supporting RECEIVED_POWER_WITH_GRANT measurement reports.

8.24.2 Test date

Start date	May 22, 2019

8.24.3 Observations, settings and special notes

None

8.24.4 Test data

Table 8.24-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT has successfully registered with SAS Test Harness, with cbsdId=C and measCapability = “RECEIVED_POWER_WITH_GRANT” • UUT has received a valid grant with grantId = G • UUT is in Grant State AUTHORIZED and is actively transmitting within the bounds of its grant. • Grant has heartbeatInterval = 60 seconds	–	–
2	UUT sends a Heartbeat Request message. Verify Heartbeat Request message contains all required parameters properly formatted, and specifically: • cbsdId = C • grantId = G • operationState = “AUTHORIZED”	☒	☐
3	SAS Test Harness sends a Heartbeat Response message, containing all required parameters properly formatted, and specifically: • cbsdId = C • grantId = G • measReportConfig= “RECEIVED_POWER_WITH_GRANT” • responseCode = 0	–	–
4	UUT sends a Heartbeat Request message. Verify message contains all required parameters properly formatted, and specifically: • cbsdId = C • grantId = G • operationState = “AUTHORIZED”	☒	☐

Step	Test Execution Steps	Pass	Fail
5	If Heartbeat Request message (step 4) contains measReport object, then: • verify measReport is properly formatted as object rcvdPowerMeasReport • end test, with PASS result else, if Heartbeat Request message (step 4) does not contain measReport object, then: • If number of Heartbeat Requests sent by UUT after Step 3 is = 5, then stop test with result of FAIL	☒	☐
6	SAS Test Harness sends a Heartbeat Response message, containing all required parameters properly formatted, and specifically: • cbsdId = C • grantId = G • responseCode = 0 Go to Step 4, above	–	–

For the test log please refer to Section 9 of this test report.

8.25 6.6.4.1.1 [WINNF.FT.C.RLQ.1] Successful Relinquishment

8.25.1 Definitions and limits

6.6 CBSD Relinquishment Process
This section provides test steps, condition and procedures to test the conformance of the CBSD implementation for the CBSD Relinquishment Procedure. A precondition is the CBSD has successfully discovered the SAS it wants to communicate with.
Each test generates a CBSD relinquishment request and validates the CBSD takes the appropriate action following the SAS relinquishment response. The CBSD shall send the Relinquishment request message after stopping the RF transmission.
Successful Relinquishment Request (responseCode 0)

8.25.2 Test date

Start date	May 22, 2019

8.25.3 Observations, settings and special notes

None

8.25.4 Test data

Table 8.25-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT has successfully registered with SAS Test Harness, with cbsdId=C • UUT has received a valid grant with grantId = G • UUT is in Grant State AUTHORIZED and is actively transmitting within the bounds of its grant. Invoke trigger to relinquish UUT Grant from the SAS Test Harness	–	–
2	UUT sends a Relinquishment Request message. Verify message contains all required parameters properly formatted, and specifically: • cbsdId = C • grantId = G	☒	☐
3	SAS Test Harness shall approve the request with a Relinquishment Response message with parameters: ○ cbsdId = C ○ grantId = G ○ responseCode = 0	–	–
4	After completion of step 3, SAS Test Harness will not provide any additional positive response (responseCode=0) to further request messages from the UUT.	–	–
5	Monitor the RF output of the UUT from start of test until 60 seconds after Step 3 is complete. This is the end of the test. Verify: • UUT shall stop RF transmission at any time between triggering the relinquishment and UUT sending the relinquishment request	☒	☐

For the test log please refer to Section 9 of this test report.

8.26 6.6.4.1.3 [WINNF.FT.C.RLQ.3] Unsuccessful Relinquishment, responseCode=102

8.26.1 Definitions and limits

6.6 CBSD Relinquishment Process
This section provides test steps, condition and procedures to test the conformance of the CBSD implementation for the CBSD Relinquishment Procedure. A precondition is the CBSD has successfully discovered the SAS it wants to communicate with.
Each test generates a CBSD relinquishment request and validates the CBSD takes the appropriate action following the SAS relinquishment response. The CBSD shall send the Relinquishment request message after stopping the RF transmission.
CBSD under test cannot be expected to generate a message with a missing or invalid parameter. To test for responseCode not equal to 0, the SAS Test Harness will respond to a message with a non-zero responseCode.
The following are the test execution steps where the Relinquishment response contains responseCode (R) = 102.

8.26.2 Test date

Start date	May 22, 2019

8.26.3 Observations, settings and special notes

None

8.26.4 Test data

Table 8.26-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT has successfully registered with SAS Test Harness, with cbsdId=C • UUT has received a valid grant with grantId = G • UUT is in Grant State AUTHORIZED and is actively transmitting within the bounds of its grant. Invoke trigger to Relinquish UUT Grant from the SAS Test Harness	–	–
2	UUT sends a Relinquishment Request message. Verify message contains all required parameters properly formatted, and specifically: • cbsdId = C • grantId = G	–	–
3	SAS Test Harness shall send a Relinquishment Response message with parameters: • cbsdId = C • No grantId • responseCode = R	–	–
4	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–
5	Monitor the RF output of the UUT from start of test until 60 seconds after Step 3 is complete. This is the end of the test. Verify: • UUT stopped RF transmission at any time between triggering the relinquishment and UUT sending the relinquishment request	☒	☐

For the test log please refer to Section 9 of this test report.

8.27 6.6.4.3.1 [WINNF.FT.C.RLQ.5] Unsuccessful Relinquishment, responseCode=103

8.27.1 Definitions and limits

6.6 CBSD Relinquishment Process
This section provides test steps, condition and procedures to test the conformance of the CBSD implementation for the CBSD Relinquishment Procedure. A precondition is the CBSD has successfully discovered the SAS it wants to communicate with.
Each test generates a CBSD relinquishment request and validates the CBSD takes the appropriate action following the SAS relinquishment response. The CBSD shall send the Relinquishment request message after stopping the RF transmission.
CBSD under test cannot be expected to generate a message with a missing or invalid parameter. To test for responseCode not equal to 0, the SAS Test Harness will respond to a message with a non-zero responseCode.
The same steps provided for WINNF.FT.C.RLQ.3 shall be executed for this test, with the exception that the Relinquishment response contains responseCode (R) = 103 and responseData = “grantId”.

8.27.2 Test date

Start date	May 22, 2019

8.27.3 Observations, settings and special notes

None

8.27.4 Test data

Table 8.27-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT has successfully registered with SAS Test Harness, with cbsdId=C • UUT has received a valid grant with grantId = G • UUT is in Grant State AUTHORIZED and is actively transmitting within the bounds of its grant. Invoke trigger to Relinquish UUT Grant from the SAS Test Harness	–	–
2	UUT sends a Relinquishment Request message. Verify message contains all required parameters properly formatted, and specifically: • cbsdId = C • grantId = G	–	–
3	SAS Test Harness shall send a Relinquishment Response message with parameters: • cbsdId = C • responseData = grantId • responseCode = 103	–	–
4	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–
5	Monitor the RF output of the UUT from start of test until 60 seconds after Step 3 is complete. This is the end of the test. Verify: • UUT stopped RF transmission at any time between triggering the relinquishment and UUT sending the relinquishment request	☒	☐

For the test log please refer to Section 9 of this test report.

8.28 6.7.4.1.1 [WINNF.FT.C.DRG.1] Successful Deregistration

8.28.1 Definitions and limits

6.7 CBSD Deregistration Process
This section explains test steps/condition/procedure for the CBSD Deregistration Request and its subsequent actions following the reception of the Deregistration Responses from the SAS.
A Deregistration request is issued by a CBSD to request a SAS to deregister the CBSD from the SAS. A Deregistration Request Message issued by a CBSD is provided in [n.5], Section 10.11.
In the Deregistration Response message, the SAS should echo back an array of DeregistrationResponse object. Each deregistrationResponse object consists of a cbsdId and a responseCode. If the deregistration request was successful, the responseCode should be set to 0, otherwise responseCode is set to appropriate error value. The deregistrationResponse Message and the deregistrationResponse object are provided in [n.5], Section 10.12.
Each test generates a CBSD deregistration request and validates the CBSD takes the appropriate actions following the SAS deregistration response.
These deregistration test cases assume the CBSD is the source (operator initiated, for instance reset site). Deregistrations triggered by the SAS in a response message with a responseCode of 105 are covered in other test cases.
A Deregistration request is issued by a CBSD to request a SAS to deregister the CBSD from the SAS. A Deregistration Request Message issued by a CBSD.
In the Deregistration Response message, the SAS should echo back an array of DeregistrationResponse object. Each deregistrationResponse object consists of a cbsdId and a responseCode. If the deregistration request was successful, the responseCode should be set to 0, otherwise responseCode is set to appropriate error value.
Each test generates a CBSD deregistration request and validates the CBSD takes the appropriate actions following the SAS deregistration response.
These deregistration test cases assume the CBSD is the source (operator initiated, for instance reset site). Deregistrations triggered by the SAS in a response message with a responseCode of 105 are covered in other test cases.
Successful Deregistration Request (responseCode 0)

8.28.2 Test date

Start date	May 22, 2019

8.28.3 Observations, settings and special notes

None

8.28.4 Test data

Table 8.28-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT has successfully registered with SAS Test Harness, with cbsdId=C • UUT has received a valid grant with grantId = G • UUT is in Grant State AUTHORIZED and is actively transmitting within the bounds of its grant. Invoke trigger to deregister UUT from the SAS Test Harness	–	–
2	UUT sends a Relinquishment request and receives Relinquishment response with responseCode=0	–	–
3	UUT sends Deregistration Request to SAS Test Harness with cbsdId = C.	☒	☐
4	SAS Test Harness shall approve the request with a Deregistration Response message with parameters: • cbsdId = C • responseCode = 0	–	–
5	After completion of step 3, SAS Test Harness will not provide any additional positive response (responseCode=0) to further request messages from the UUT.	–	–

Step	Test Execution Steps	Pass	Fail
6	Monitor the RF output of the UUT from start of test until 60 seconds after Step 4 is complete. This is the end of the test. Verify: • UUT stopped RF transmission at any time between triggering the deregistration and either A OR B occurs: A. UUT sending a Registration Request message, as this is not mandatory B. UUT sending a Deregistration Request message	☒	☐

For the test log please refer to Section 9 of this test report.

8.29 6.7.4.2.1 [WINNF.FT.C.DRG.3] Deregistration responseCode=102

8.29.1 Definitions and limits

6.7 CBSD Deregistration Process
This section explains test steps/condition/procedure for the CBSD Deregistration Request and its subsequent actions following the reception of the Deregistration Responses from the SAS.
A Deregistration request is issued by a CBSD to request a SAS to deregister the CBSD from the SAS. A Deregistration Request Message issued by a CBSD is provided in [n.5], Section 10.11.
In the Deregistration Response message, the SAS should echo back an array of DeregistrationResponse object. Each deregistrationResponse object consists of a cbsdId and a responseCode. If the deregistration request was successful, the responseCode should be set to 0, otherwise responseCode is set to appropriate error value. The deregistrationResponse Message and the deregistrationResponse object are provided in [n.5], Section 10.12.
Each test generates a CBSD deregistration request and validates the CBSD takes the appropriate actions following the SAS deregistration response.
These deregistration test cases assume the CBSD is the source (operator initiated, for instance reset site). Deregistrations triggered by the SAS in a response message with a responseCode of 105 are covered in other test cases.
A Deregistration request is issued by a CBSD to request a SAS to deregister the CBSD from the SAS. A Deregistration Request Message issued by a CBSD.
In the Deregistration Response message, the SAS should echo back an array of DeregistrationResponse object. Each deregistrationResponse object consists of a cbsdId and a responseCode. If the deregistration request was successful, the responseCode should be set to 0, otherwise responseCode is set to appropriate error value.
Each test generates a CBSD deregistration request and validates the CBSD takes the appropriate actions following the SAS deregistration response.
These deregistration test cases assume the CBSD is the source (operator initiated, for instance reset site). Deregistrations triggered by the SAS in a response message with a responseCode of 105 are covered in other test cases.
CBSD under test cannot be expected to generate a message with a missing or invalid parameter. To test for responseCode not equal to 0, the SAS Test Harness will respond to a message with a non-zero responseCode. The following are the test execution steps where the Deregistration response contains responseCode (R) = 102.

8.29.2 Test date

Start date	May 22, 2019

8.29.3 Observations, settings and special notes

None

8.29.4 Test data

Table 8.29-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT has successfully registered with SAS Test Harness, with cbsdId=C • UUT has received a valid grant with grantId = G • UUT is in Grant State AUTHORIZED and is actively transmitting within the bounds of its grant. Invoke trigger to deregister UUT from the SAS Test Harness	–	–
2	UUT sends a Relinquishment request and receives Relinquishment response with responseCode=0	–	–
3	UUT sends Deregistration Request to SAS Test Harness with cbsdId = C	–	–
4	The SAS Test Harness sends the Deregistration Response Message to UUT with: • No cbsdId • responseCode = 102	–	–
5	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–

Step	Test Execution Steps	Pass	Fail
6	Monitor the RF output of the UUT from start of test until 60 seconds after Step 4 is complete. This is the end of the test. Verify: • UUT stopped RF transmission at any time between triggering the deregistration and either A OR B occurs: A. UUT sending a Registration Request message, as this is not mandatory B. UUT sending a Deregistration Request message	☒	☐

For the test log please refer to Section 9 of this test report.

8.30 6.7.4.3.1 [WINNF.FT.C.DRG.5] Deregistration responseCode=103

8.30.1 Definitions and limits

6.7 CBSD Deregistration Process
This section explains test steps/condition/procedure for the CBSD Deregistration Request and its subsequent actions following the reception of the Deregistration Responses from the SAS.
A Deregistration request is issued by a CBSD to request a SAS to deregister the CBSD from the SAS. A Deregistration Request Message issued by a CBSD is provided in [n.5], Section 10.11.
In the Deregistration Response message, the SAS should echo back an array of DeregistrationResponse object. Each deregistrationResponse object consists of a cbsdId and a responseCode. If the deregistration request was successful, the responseCode should be set to 0, otherwise responseCode is set to appropriate error value. The deregistrationResponse Message and the deregistrationResponse object are provided in [n.5], Section 10.12.
Each test generates a CBSD deregistration request and validates the CBSD takes the appropriate actions following the SAS deregistration response.
These deregistration test cases assume the CBSD is the source (operator initiated, for instance reset site). Deregistrations triggered by the SAS in a response message with a responseCode of 105 are covered in other test cases.
A Deregistration request is issued by a CBSD to request a SAS to deregister the CBSD from the SAS. A Deregistration Request Message issued by a CBSD.
In the Deregistration Response message, the SAS should echo back an array of DeregistrationResponse object. Each deregistrationResponse object consists of a cbsdId and a responseCode. If the deregistration request was successful, the responseCode should be set to 0, otherwise responseCode is set to appropriate error value.
Each test generates a CBSD deregistration request and validates the CBSD takes the appropriate actions following the SAS deregistration response.
These deregistration test cases assume the CBSD is the source (operator initiated, for instance reset site). Deregistrations triggered by the SAS in a response message with a responseCode of 105 are covered in other test cases.
CBSD under test cannot be expected to generate a message with a missing or invalid parameter. To test for responseCode not equal to 0, the SAS Test Harness will respond to a message with a non-zero responseCode. The same steps provided for WINNF.FT.C.DRG.3 shall be executed for this test, with the exception that the Deregistration response contains responseCode (R) = 103 and responseData = “cbsdId”.

8.30.2 Test date

Start date	May 22, 2019

8.30.3 Observations, settings and special notes

None

8.30.4 Test data

Table 8.30-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with SAS Test Harness • UUT has successfully registered with SAS Test Harness, with cbsdId=C • UUT has received a valid grant with grantId = G • UUT is in Grant State AUTHORIZED and is actively transmitting within the bounds of its grant. Invoke trigger to deregister UUT from the SAS Test Harness	–	–
2	UUT sends a Relinquishment request and receives Relinquishment response with responseCode=0	–	–
3	UUT sends Deregistration Request to SAS Test Harness with cbsdId = C	–	–
4	The SAS Test Harness sends the Deregistration Response Message to UUT with: • responseData = cbsdId • responseCode = 103	–	–
5	After completion of step 3, SAS Test Harness will not provide any positive response (responseCode=0) to further request messages from the UUT.	–	–

Step	Test Execution Steps	Pass	Fail
6	Monitor the RF output of the UUT from start of test until 60 seconds after Step 4 is complete. This is the end of the test. Verify: • UUT stopped RF transmission at any time between triggering the deregistration and either A OR B occurs: A. UUT sending a Registration Request message, as this is not mandatory B. UUT sending a Deregistration Request message	☒	☐

For the test log please refer to Section 9 of this test report.

8.31 6.8.4.1.1 [WINNF.FT.C.SCS.1] Successful TLS connection between UUT and SAS Test Harness

8.31.1 Definitions and limits

6.8 CBSD Security Validation
This section provides test steps, condition and procedures to test the conformance of the CBSD implementation for the Security Establishment Procedure. A precondition is the CBSD has successfully discovered the SAS it wants to communicate with.
In all test cases under this category, the TLS connection is established successfully between the SAS Test Harness and CBSD. A pre-condition for these tests is that Certificates at CBSD and SAS Test Harness are correct and valid. The security procedure is irrespective of the procedures defined for the SAS Test Harness to CBSD communication.

8.31.2 Test date

Start date	October 3, 2018

8.31.3 Observations, settings and special notes

Place in the WInnForum SAS Test Harness the correct SAS Test Harness X.509 certificates for this test case. Edit the conf.xml file appropriately for use of this certificate.
Verify the SAS Test Harness X.509 certificate is the correct X.509 certificate for this test case by inspecting its content as described in the "readme_file_x509_RSA_certs_test_labs.txt" [WINNF-IN-0156 Version V1.0.0.1]. For test case [WINNF.FT.C.SCS.1] the X.509 certificate is the regular SAS Test Harness X.509 certificate used for the Interface Conformance Testing in [WINNF-TS-0122 Version V1.0.0].
The method for executing CBSD/DP UUT security test case is via Wireshark.

8.31.4 Test data

Table 8.31-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Verify in Wireshark the following in the captured packets: 1. Wireshark "Protocol" column shows "TLSv1.2" 2. CBSD/DP UUT sends "Client Hello" message to WInnForum SAS Test Harness WInnForum SAS Test Harness sends "Server Hello" message to CBSD/DP UUT. • The "Server Hello" message "Handshake Protocol" IE includes the "Cipher Suite" IE. • Verify the "Cipher Suite" shown in Wireshark is one of the following: TLS_RSA_WITH_AES_128_GCM_SHA256, TLS_RSA_WITH_AES_256_GCM_SHA384, TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256, TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384, TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 3. "Application Data" messages are exchanged between WInnForum SAS Test Harness and CBSD/DP UUT.	☒	☐
2	Verify that WInnForum SAS Test Harness Command Prompt shows Registration Request Message from CBSD/DP UUT	☒	☐

For the test log please refer to Section 9 of this test report.

8.32 6.8.4.2.1 [WINNF.FT.C.SCS.2] TLS failure due to revoked certificate

8.32.1 Definitions and limits

This section provides test steps, condition and procedures to test the conformance of the CBSD implementation for the Security Establishment Procedure. A precondition is the CBSD has successfully discovered the SAS it wants to communicate with.
In all test cases under this category, the TLS connection is not established successfully between the SAS Test Harness and CBSD. The security procedure is irrespective of the procedures defined for the SAS Test Harness to CBSD communication.
Test case pre-requisite:
• The certificate at the SAS Test Harness shall be marked as revoked

8.32.2 Test date

Start date	October 3, 2018

8.32.3 Observations, settings and special notes

Place in the WInnForum SAS Test Harness the correct SAS Test Harness X.509 certificates for this test case. Edit the conf.xml file appropriately for use of this certificate.
Verify the SAS Test Harness X.509 certificate is the correct X.509 certificate for this test case by inspecting its content as described in the "readme_file_x509_RSA_certs_test_labs.txt" [WINNF-IN-0156 Version V1.0.0.1]. For test case [WINNF.FT.C.SCS.2] the X.509 certificate has

- Proper Validity time (the X.509 certificate is not expired)
- X.509v3 extension of "Authority Information Access: OCSP - URI: http://ocsp.testharness.cbrstestlab.com" (this URI is an example of the OCSP server available for the test lab)
- X.509v3 extension of "CRL Distribution Points: Full Name: URI: http://crlserver.testharness.cbrstestlab.com/crlserver.crl" (this URI is an example of the CRL server and CRL file available for the test lab)
- Certificate Serial Number appears as "Revoked" in the CRL file located in the CRL server available for the test lab, or appears as "Revoked" in the OCSP server available for the test lab.

For execution of this test case the CRL file must have proper validity. If this test is intended to be executed when the validity date of the CRL file has expired, a new CRL file with proper validity needs to be generated as described in the "readme_file_x509_RSA_certs_test_labs.txt" [WINNF-IN-0156 Version V1.0.0.1].
For execution of this test case, the test lab also requires an available DNS server to resolve FQDNs of the OCSP server or CRL server.
The method for executing CBSD/DP UUT security test case is via Wireshark.

8.32.4 Test data

Table 8.32-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Verify in Wireshark the following in the captured packets: 1. Wireshark "Protocol" column shows "TLSv1.2" 2. CBSD/DP UUT sends "Client Hello" message to WInnForum SAS Test Harness 3. WInnForum SAS Test Harness sends "Server Hello" message to CBSD/DP UUT. • The "Server Hello" message "Handshake Protocol" IE includes the "Cipher Suite" IE. • Verify the "Cipher Suite" shown in Wireshark is one of the following: TLS_RSA_WITH_AES_128_GCM_SHA256, TLS_RSA_WITH_AES_256_GCM_SHA384, TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256, TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384, TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 4. CBSD/DP UUT performs DNS resolution for the FQDN of the CRL server, or OCSP server, or both listed in the X.509v3 extensions described above for the X.509 certificate of SAS Test Harness. 5. CBSD/DP UUT: • Download the CRL file according to the full URI listed in X.509v3 extension of "CRL Distribution Points" described above. OR • Send to the OCSP server an OCSP "Request" message containing the certificate serial number, and OCSP server replies. OR • Both CRL file download and OCSP transaction as described above. 6. "Application Data" messages are not seen between WInnForum SAS Test Harness and CBSD/DP UUT. 7. CBSD/DP UUT may send a TLS "Alert" message to WInnForum SAS Test Harness notifying of rejecting the TLS connection before attempting to establish the TLS connection again.	☒	☐
2	Verify that WInnForum SAS Test Harness Command Prompt does not show any Request Message from CBSD/DP UUT	☒	☐

For the test log please refer to Section 9 of this test report.

8.33 6.8.4.2.2 [WINNF.FT.C.SCS.3] TLS failure due to expired server certificate

8.33.1 Definitions and limits

This section provides test steps, condition and procedures to test the conformance of the CBSD implementation for the Security Establishment Procedure. A precondition is the CBSD has successfully discovered the SAS it wants to communicate with.
In all test cases under this category, the TLS connection is not established successfully between the SAS Test Harness and CBSD. The security procedure is irrespective of the procedures defined for the SAS Test Harness to CBSD communication.
Test case pre-requisite:
• Configure the SAS Test Harness such that server certificate is valid but expired.

8.33.2 Test date

Start date	October 3, 2018

8.33.3 Observations, settings and special notes

Place in the WInnForum SAS Test Harness the correct SAS Test Harness X.509 certificates for this test case. Edit the conf.xml file appropriately for use of this certificate.
Verify the SAS Test Harness X.509 certificate is the correct X.509 certificate for this test case by inspecting its content as described in the "readme_file_x509_RSA_certs_test_labs.txt" [WINNF-IN-0156 Version V1.0.0.1]. For test case [WINNF.FT.C.SCS.3] the X.509 certificate has
• Expired Validity time. The date appearing in the "Not After" parameter of the X.509 certificate has passed.
The method for executing CBSD/DP UUT security test case is via Wireshark.

8.33.4 Test data

Table 8.33-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Verify in Wireshark the following in the captured packets: 1. Wireshark "Protocol" column shows "TLSv1.2" 2. CBSD/DP UUT sends "Client Hello" message to WInnForum SAS Test Harness 3. WInnForum SAS Test Harness sends "Server Hello" message to CBSD/DP UUT. • The "Server Hello" message "Handshake Protocol" IE includes the "Cipher Suite" IE. • Verify the "Cipher Suite" shown in Wireshark is one of the following: TLS_RSA_WITH_AES_128_GCM_SHA256, TLS_RSA_WITH_AES_256_GCM_SHA384, TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256, TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384, TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 4. "Application Data" messages are exchanged between WInnForum SAS Test Harness and CBSD/DP UUT.	☒	☐
2	Verify that WInnForum SAS Test Harness Command Prompt does not show any Request Message from CBSD/DP UUT	☒	☐

For the test log please refer to Section 9 of this test report.

8.34 6.8.4.2.3 [WINNF.FT.C.SCS.4] TLS failure when SAS Test Harness certificate is issued by an unknown CA

8.34.1 Definitions and limits

This section provides test steps, condition and procedures to test the conformance of the CBSD implementation for the Security Establishment Procedure. A precondition is the CBSD has successfully discovered the SAS it wants to communicate with.
In all test cases under this category, the TLS connection is not established successfully between the SAS Test Harness and CBSD. The security procedure is irrespective of the procedures defined for the SAS Test Harness to CBSD communication.
Test case pre-requisite:
• Equip the SAS Test Harness with certificate signed by an unknown CA to the CBSD.

8.34.2 Test date

Start date	October 3, 2018

8.34.3 Observations, settings and special notes

Place in the WInnForum SAS Test Harness the correct SAS Test Harness X.509 certificates for this test case. Edit the conf.xml file appropriately for use of this certificate.
Verify the SAS Test Harness X.509 certificate is the correct X.509 certificate for this test case by inspecting its content as described in the "readme_file_x509_RSA_certs_test_labs.txt" [WINNF-IN-0156 Version V1.0.0.1]. For test case [WINNF.FT.C.SCS.4] the X.509 certificate has

- PKI chain which is not known to the CBSD/DP UUT, and is different from the PKI chain of the SAS Test Harness X.509 certificate used in test WINNF.FT.C.SCS.1.

The method for executing CBSD/DP UUT security test case is via Wireshark.

8.34.4 Test data

Table 8.34-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Verify in Wireshark the following in the captured packets: 1. Wireshark "Protocol" column shows "TLSv1.2" 2. CBSD/DP UUT sends "Client Hello" message to WInnForum SAS Test Harness 3. WInnForum SAS Test Harness sends "Server Hello" message to CBSD/DP UUT. • The "Server Hello" message "Handshake Protocol" IE includes the "Cipher Suite" IE. • Verify the "Cipher Suite" shown in Wireshark is one of the following: TLS_RSA_WITH_AES_128_GCM_SHA256, TLS_RSA_WITH_AES_256_GCM_SHA384, TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256, TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384, TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 4. "Application Data" messages are not seen between WInnForum SAS Test Harness and CBSD/DP UUT. 5. CBSD/DP UUT may send a TLS "Alert" message to WInnForum SAS Test Harness notifying of rejecting the TLS connection before attempting to establish the TLS connection again.	☒	☐
2	Verify that WInnForum SAS Test Harness Command Prompt does not show any Request Message from CBSD/DP UUT	☒	☐

For the test log please refer to Section 9 of this test report.

8.35 6.8.4.2.4 [WINNF.FT.C.SCS.5] TLS failure when certificate at the SAS Test Harness is corrupted

8.35.1 Definitions and limits

This section provides test steps, condition and procedures to test the conformance of the CBSD implementation for the Security Establishment Procedure. A precondition is the CBSD has successfully discovered the SAS it wants to communicate with.
In all test cases under this category, the TLS connection is not established successfully between the SAS Test Harness and CBSD. The security procedure is irrespective of the procedures defined for the SAS Test Harness to CBSD communication.
Test case pre-requisite:
• The end-entity certificate at the SAS Test Harness shall be corrupted

8.35.2 Test date

Start date	October 3, 2018

8.35.3 Observations, settings and special notes

Place in the WInnForum SAS Test Harness the correct SAS Test Harness X.509 certificates for this test case. Edit the conf.xml file appropriately for use of this certificate.
Verify the SAS Test Harness X.509 certificate is the correct X.509 certificate for this test case by inspecting its content as described in the "readme_file_x509_RSA_certs_test_labs.txt" [WINNF-IN-0156 Version V1.0.0.1]. For test case [WINNF.FT.C.SCS.5] the X.509 certificate has
• Invalid Signature as described in the "readme_file_x509_RSA_certs_test_labs.txt" [WINNF-IN-0156 Version V1.0.0.1].
The method for executing CBSD/DP UUT security test case is via Wireshark.

8.35.4 Test data

Table 8.35-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Verify in Wireshark the following in the captured packets: 1. Wireshark "Protocol" column shows "TLSv1.2" 2. CBSD/DP UUT sends "Client Hello" message to WInnForum SAS Test Harness 3. WInnForum SAS Test Harness sends "Server Hello" message to CBSD/DP UUT. • The "Server Hello" message "Handshake Protocol" IE includes the "Cipher Suite" IE. • Verify the "Cipher Suite" shown in Wireshark is one of the following: TLS_RSA_WITH_AES_128_GCM_SHA256, TLS_RSA_WITH_AES_256_GCM_SHA384, TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256, TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384, TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 4. "Application Data" messages are not seen between WInnForum SAS Test Harness and CBSD/DP UUT. 5. CBSD/DP UUT may send a TLS "Alert" message to WInnForum SAS Test Harness notifying of rejecting the TLS connection before attempting to establish the TLS connection again.	☒	☐
2	Verify that WInnForum SAS Test Harness Command Prompt does not show any Request Message from CBSD/DP UUT	☒	☐

For the test log please refer to Section 9 of this test report.

8.36 7.1.4.1.1 [WINNF.PT.C.HBT] UUT RF Transmit Power Measurement

8.36.1 Definitions and limits

This section provides test steps, condition and procedures to demonstrate conformance of the CBSD to limitations on transmit power due to maxEirp setting of AUTHORIZED grants for that CBSD.

8.36.1 Test date

Start date	May 21, 2019

8.36.1 Observations, settings and special notes

None

8.36.2 Test data

Table 8.36-1: *Test results*

Step	Test Execution Steps	Pass	Fail
1	Ensure the following conditions are met for test entry: • UUT has successfully completed SAS Discovery and Authentication with the SAS Test Harness • UUT has registered with the SAS, with CBSD ID = C • UUT has a single valid grant G with parameters {lowFrequency = FL, highFrequency = FH, maxEirp = Pi}, with grant in AUTHORIZED state, and grantExpireTime set to a value far past the duration of this test case	–	–
2	UUT and SAS Test Harness perform a series of Heartbeat Request/Response cycles, which continues until the other test steps are complete. Messaging for each cycle is as follows: • UUT sends Heartbeat Request, including: o cbsdId = C o grantId = G • SAS Test Harness responds with Heartbeat Response, including: o cbsdId = C o grantId = G o transmitExpireTime = current UTC time + 200 seconds o responseCode = 0	–	–
3	Tester performs power measurement on RF interface(s) of UUT, and verifies it complies with the maxEirp setting, Pi. The RF measurement method is out of scope of this document, but may include additional configuration of the UUT, as required, to fulfil the requirements of the power measurement method.	☒	☐

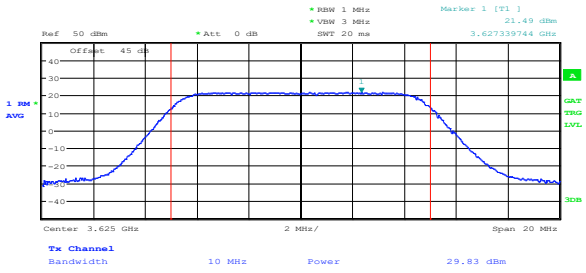

Figure 8.36-1: *Output power and power density validation when maxEirp setting Pi = 29.7*

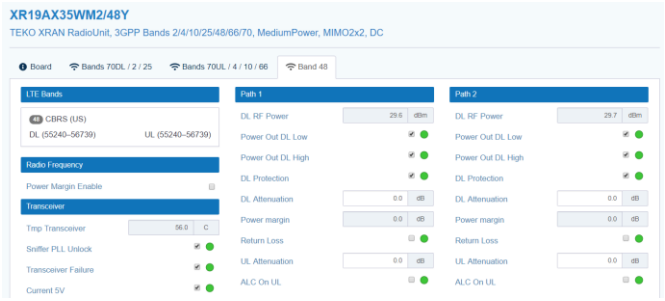

Figure 8.36-2: *Setting of the power on GUI*

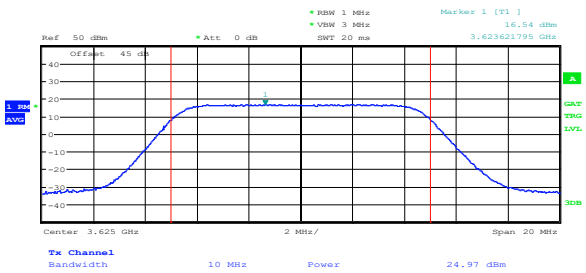

Figure 8.36-3: *Output power and power density validation when maxEirp setting Pi = 24.9 (reduced power)*

Figure 8.36-4: *Setting of the power on GUI (reduced power)*

Section 9. Log files library

9.1 Log file for test case ID: WINNF.FT.C.REG.1


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [],
   "userId": "abc"
  }
 ]
}
<7>17:18:48.431 Sas.cpp          post             06806 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
```

9.2 Log file for test case ID: WINNF.FT.C.REG.3


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:24:24.877 Sas.cpp          post             07389 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
```

9.3 Log file for test case ID: WINNF.FT.C.REG.7


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:32:14.045 Sas.cpp post 07872 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>17:32:14.046 CbrsDaemon.cpp onLoop 07872 [34;1mINF[0m Listening for 22 seconds
<7>17:32:14.047 SpvLaunchdProxy.cpp create 07872 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>17:32:14.047 SpvLaunchdProxy.cpp create 07872 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>17:32:14.047 SpvLaunchdProxy.cpp initSpvLaunchdProxy 07872 [36;1mDBG[0m SpvLaunchd is
running.
<7>17:32:14.047 SpvLaunchdProxy.cpp logDBusMessage 07872 [36;1mDBG[0m handleRequest:
signal sender=org.freedesktop.DBus -> dest=:1.33 serial=2 path=/org/freedesktop/DBus;
interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>17:32:14.047 SpvLaunchdProxy.cpp dbusHandler 07872 [36;1mDBG[0m NameAcquired:
:1.33
<7>17:32:14.047 SpvLaunchdProxy.cpp dbusHandler 07872 [36;1mDBG[0m Connection name:
:1.33
<6>17:32:15.070 CbrsDaemon.cpp parseTree 07872 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<6>17:32:15.072 CbrsDaemon.cpp onLoop 07872 [34;1mINF[0m Listening for 22 seconds
<6>17:32:38.117 CbrsDaemon.cpp parseTree 07872 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<6>17:32:38.119 Cbsd.cpp cbsd_main_ 07872 [34;1mINF[0m All grants belonging to XM2-
X19AX35M2Mock-SAS1012482003, initialized after change in CPRI device configuration
<6>17:32:38.119 Cbsd.cpp cbsd_main_ 07872 [34;1mINF[0m Deregistration procedure for
CBSD XM2-X19AX35M2Mock-SAS1012482003
<7>17:32:38.119 Sas.cpp post 07872 [36;1mDBG[0m {
 "deregistrationRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003"
  }
 ]
}
<7>17:32:38.122 Sas.cpp post 07872 [36;1mDBG[0m {
 "deregistrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>17:32:38.122 Sas.cpp post 07872 [36;1mDBG[0m {
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 16.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:32:38.162 Sas.cpp post 07872 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 200
   }
  }
 ]
}
<6>17:32:38.163 CbrsDaemon.cpp onLoop 07872 [34;1mINF[0m Listening for 22
seconds
<6>17:33:01.208 CbrsDaemon.cpp parseTree 07872 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>17:33:01.210 Sas.cpp post 07872 [36;1mDBG[0m {
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
```

```
    },
    "callSign": "?",
    "cbsdCategory": "A",
    "cbsdInfo": {
     "firmwareVersion": "v2.0.5",
     "hardwareVersion": "v1.0.45",
     "model": "CPRI_DEVICE-XXX",
     "softwareVersion": "v1.2.1",
     "vendor": "JMA Wireless"
    },
    "cbsdSerialNumber": "1012482003",
    "fccId": "XM2-X19AX35M2",
    "installationParam": {
     "antennaAzimuth": 70,
     "antennaBeamwidth": 45,
     "antennaDowntilt": 36,
     "antennaGain": 0,
     "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
     "eirpCapability": 15,
     "height": 16.0,
     "heightType": "AMSL",
     "horizontalAccuracy": 49,
     "indoorDeployment": true,
     "latitude": 43.09,
     "longitude": -76.15,
     "verticalAccuracy": 2
    },
    "measCapability": [
     "RECEIVED_POWER_WITH_GRANT"
    ],
    "userId": "abc"
   }
  ]
}
<7>17:33:01.211 Sas.cpp          post            07872 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 200
   }
  }
 ]
}
<6>17:33:01.211 CbrsDaemon.cpp     onLoop          07872 [34;1mINF[0m Listening for 22 seconds
<6>17:33:24.256 CbrsDaemon.cpp     parseTree       07872 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>17:33:24.258 Sas.cpp          post            07872 [36;1mDBG[0m {
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 16.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:33:24.259 Sas.cpp          post            07872 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 200
   }
  }
 ]
}
```

9.4 Log file for test case ID: WINNF.FT.C.REG.8


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:36:58.192 Sas.cpp post 08064 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 102
   }
  }
 ]
}
<7>17:36:58.192 Cbsd.cpp cbsd_main_ 08064 [36;1mDBG[0m ERROR state reset to
UNREGISTERED
<6>17:36:58.193 CbrsDaemon.cpp onLoop 08064 [34;1mINF[0m Listening for 22 seconds
<7>17:36:58.193 SpvLaunchdProxy.cpp create 08064 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>17:36:58.193 SpvLaunchdProxy.cpp create 08064 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>17:36:58.193 SpvLaunchdProxy.cpp initSpvLaunchdProxy 08064 [36;1mDBG[0m SpvLaunchd is
running.
<7>17:36:58.194 SpvLaunchdProxy.cpp logDBusMessage 08064 [36;1mDBG[0m handleRequest:
signal sender=org.freedesktop.DBus -> dest=:1.35 serial=2 path=/org/freedesktop/DBus;
interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>17:36:58.194 SpvLaunchdProxy.cpp dbusHandler 08064 [36;1mDBG[0m NameAcquired:
:1.35
<7>17:36:58.194 SpvLaunchdProxy.cpp dbusHandler 08064 [36;1mDBG[0m Connection name:
:1.35
<6>17:36:59.216 CbrsDaemon.cpp parseTree 08064 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>17:36:59.218 Sas.cpp post 08064 [36;1mDBG[0m {
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:36:59.219 Sas.cpp post 08064 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 200
   }
  }
 ]
}
```

9.5 Log file for test case ID: WINNF.FT.C.REG.10


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:38:54.046 Sas.cpp          post            08207 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 200
   }
  }
 ]
}
<6>17:38:54.046 CbrsDaemon.cpp     onLoop          08207 [34;1mINF[0m Listening for 22 seconds
<7>17:38:54.047 SpvLaunchdProxy.cpp  create          08207 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>17:38:54.047 SpvLaunchdProxy.cpp  create          08207 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>17:38:54.047 SpvLaunchdProxy.cpp  initSpvLaunchdProxy  08207 [36;1mDBG[0m SpvLaunchd is
running.
<7>17:38:54.047 SpvLaunchdProxy.cpp  logDBusMessage    08207 [36;1mDBG[0m handleRequest:
signal sender=org.freedesktop.DBus -> dest=:1.37 serial=2 path=/org/freedesktop/DBus;
interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>17:38:54.047 SpvLaunchdProxy.cpp  dbusHandler       08207 [36;1mDBG[0m NameAcquired:
:1.37
<7>17:38:54.047 SpvLaunchdProxy.cpp  dbusHandler       08207 [36;1mDBG[0m Connection name:
:1.37
<6>17:38:55.070 CbrsDaemon.cpp     parseTree       08207 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>17:38:55.072 Sas.cpp          post            08207 [36;1mDBG[0m {
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:38:55.073 Sas.cpp          post            08207 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 200
   }
  }
 ]
}
```

9.6 Log file for test case ID: WINNF.FT.C.REG.12


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:40:06.825 Sas.cpp          post            08338 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 103
   }
  }
 ]
}
<7>17:40:06.825 Cbsd.cpp         cbsd_main_      08338 [36;1mDBG[0m ERROR state reset to
UNREGISTERED
<6>17:40:06.825 CbrsDaemon.cpp   onLoop          08338 [34;1mINF[0m Listening for 22 seconds
<7>17:40:06.826 SpvLaunchdProxy.cpp  create       08338 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>17:40:06.826 SpvLaunchdProxy.cpp  create       08338 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>17:40:06.826 SpvLaunchdProxy.cpp  initSpvLaunchdProxy  08338 [36;1mDBG[0m SpvLaunchd is
running.
<7>17:40:06.826 SpvLaunchdProxy.cpp  logDBusMessage  08338 [36;1mDBG[0m handleRequest:
signal sender=org.freedesktop.DBus -> dest=:1.39 serial=2 path=/org/freedesktop/DBus;
interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>17:40:06.826 SpvLaunchdProxy.cpp  dbusHandler  08338 [36;1mDBG[0m NameAcquired:
:1.39
<7>17:40:06.826 SpvLaunchdProxy.cpp  dbusHandler  08338 [36;1mDBG[0m Connection name:
:1.39
<6>17:40:07.849 CbrsDaemon.cpp   parseTree       08338 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>17:40:07.851 Sas.cpp          post            08338 [36;1mDBG[0m {
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:40:07.852 Sas.cpp          post            08338 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 200
   }
  }
 ]
}
```

9.7 Log file for test case ID: WINNF.FT.C.REG.14


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:41:10.422 Sas.cpp          post            08360 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 101
   }
  }
 ]
}
<7>17:41:10.422 Cbsd.cpp         cbsd_main_      08360 [36;1mDBG[0m ERROR state reset to
UNREGISTERED
<6>17:41:10.422 CbrsDaemon.cpp   onLoop          08360 [34;1mINF[0m Listening for 22 seconds
<7>17:41:10.423 SpvLaunchdProxy.cpp  create      08360 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>17:41:10.423 SpvLaunchdProxy.cpp  create      08360 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>17:41:10.423 SpvLaunchdProxy.cpp  initSpvLaunchdProxy  08360 [36;1mDBG[0m SpvLaunchd is
running.
<7>17:41:10.423 SpvLaunchdProxy.cpp  logDBusMessage  08360 [36;1mDBG[0m handleRequest:
signal sender=org.freedesktop.DBus -> dest=:1.40 serial=2 path=/org/freedesktop/DBus;
interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>17:41:10.423 SpvLaunchdProxy.cpp  dbusHandler     08360 [36;1mDBG[0m NameAcquired:
:1.40
<7>17:41:10.423 SpvLaunchdProxy.cpp  dbusHandler     08360 [36;1mDBG[0m Connection name:
:1.40
<6>17:41:11.446 CbrsDaemon.cpp   parseTree       08360 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>17:41:11.447 Sas.cpp          post            08360 [36;1mDBG[0m {
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:41:11.448 Sas.cpp          post            08360 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 200
   }
  }
 ]
}
```

9.8 Log file for test case ID: WINNF.FT.C.REG.16


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:42:34.342 Sas.cpp          post            08495 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 100
   }
  }
 ]
}
<7>17:42:34.342 Cbsd.cpp         cbsd_main_      08495 [36;1mDBG[0m ERROR state reset to
UNREGISTERED
<6>17:42:34.343 CbrsDaemon.cpp   onLoop          08495 [34;1mINF[0m Listening for 22 seconds
<7>17:42:34.343 SpvLaunchdProxy.cpp  create       08495 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>17:42:34.343 SpvLaunchdProxy.cpp  create       08495 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>17:42:34.343 SpvLaunchdProxy.cpp  initSpvLaunchdProxy  08495 [36;1mDBG[0m SpvLaunchd is
running.
<7>17:42:34.343 SpvLaunchdProxy.cpp  logDBusMessage  08495 [36;1mDBG[0m handleRequest:
signal sender=org.freedesktop.DBus -> dest=:1.42 serial=2 path=/org/freedesktop/DBus;
interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>17:42:34.343 SpvLaunchdProxy.cpp  dbusHandler  08495 [36;1mDBG[0m NameAcquired:
:1.42
<7>17:42:34.343 SpvLaunchdProxy.cpp  dbusHandler  08495 [36;1mDBG[0m Connection name:
:1.42
<6>17:42:35.366 CbrsDaemon.cpp   parseTree       08495 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>17:42:35.368 Sas.cpp          post            08495 [36;1mDBG[0m {
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:42:35.369 Sas.cpp          post            08495 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 200
   }
  }
 ]
}
```

9.9 Log file for test case ID: WINNF.FT.C.REG.18


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:43:48.262 Sas.cpp          post            08518 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 201
   }
  }
 ]
}
<7>17:43:48.263 Cbsd.cpp         cbsd_main_      08518 [36;1mDBG[0m ERROR state reset to
UNREGISTERED
<6>17:43:48.263 CbrsDaemon.cpp   onLoop          08518 [34;1mINF[0m Listening for 22 seconds
<7>17:43:48.264 SpvLaunchdProxy.cpp  create       08518 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>17:43:48.264 SpvLaunchdProxy.cpp  create       08518 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>17:43:48.264 SpvLaunchdProxy.cpp  initSpvLaunchdProxy  08518 [36;1mDBG[0m SpvLaunchd is
running.
<7>17:43:48.264 SpvLaunchdProxy.cpp  logDBusMessage  08518 [36;1mDBG[0m handleRequest:
signal sender=org.freedesktop.DBus -> dest=:1.43 serial=2 path=/org/freedesktop/DBus;
interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>17:43:48.264 SpvLaunchdProxy.cpp  dbusHandler  08518 [36;1mDBG[0m NameAcquired:
:1.43
<7>17:43:48.264 SpvLaunchdProxy.cpp  dbusHandler  08518 [36;1mDBG[0m Connection name:
:1.43
<6>17:43:49.287 CbrsDaemon.cpp   parseTree       08518 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>17:43:49.288 Sas.cpp          post            08518 [36;1mDBG[0m {
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:43:49.289 Sas.cpp          post            08518 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 200
   }
  }
 ]
}
```

9.10 Log file for test case ID: WINNF.FT.C.REG.20


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:48:49.866 Sas.cpp post 08946 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>17:48:49.867 CbrsDaemon.cpp onLoop 08946 [34;1mINF[0m Listening for 22 seconds
<7>17:48:49.867 SpvLaunchdProxy.cpp create 08946 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>17:48:49.867 SpvLaunchdProxy.cpp create 08946 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>17:48:49.867 SpvLaunchdProxy.cpp initSpvLaunchdProxy 08946 [36;1mDBG[0m SpvLaunchd is
running.
<7>17:48:49.868 SpvLaunchdProxy.cpp logDBusMessage 08946 [36;1mDBG[0m handleRequest:
signal sender=org.freedesktop.DBus -> dest=:1.50 serial=2 path=/org/freedesktop/DBus;
interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>17:48:49.868 SpvLaunchdProxy.cpp dbusHandler 08946 [36;1mDBG[0m NameAcquired:
:1.50
<7>17:48:49.868 SpvLaunchdProxy.cpp dbusHandler 08946 [36;1mDBG[0m Connection name:
:1.50
<6>17:48:50.890 CbrsDaemon.cpp parseTree 08946 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<6>17:48:50.892 CbrsDaemon.cpp onLoop 08946 [34;1mINF[0m Listening for 22 seconds
<6>17:49:13.923 CbrsDaemon.cpp parseTree 08946 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<6>17:49:13.926 CbrsDaemon.cpp onLoop 08946 [34;1mINF[0m Listening for 22 seconds
<6>17:49:36.971 CbrsDaemon.cpp parseTree 08946 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<6>17:49:36.973 Cbsd.cpp cbsd_main_ 08946 [34;1mINF[0m All grants belonging to XM2-
X19AX35M2Mock-SAS1012482003, initialized after change in CPRI device configuration
<6>17:49:36.973 Cbsd.cpp cbsd_main_ 08946 [34;1mINF[0m Deregistration procedure for
CBSD XM2-X19AX35M2Mock-SAS1012482003
<7>17:49:36.973 Sas.cpp post 08946 [36;1mDBG[0m {
 "deregistrationRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003"
  }
 ]
}
<7>17:49:36.976 Sas.cpp post 08946 [36;1mDBG[0m {
 "deregistrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>17:49:36.977 Sas.cpp post 08946 [36;1mDBG[0m {
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 20.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 41.0,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:49:37.017 Sas.cpp post 08946 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "response": {
    "responseCode": 200
   }
  }
 ]
}
<6>17:49:37.018 CbrsDaemon.cpp onLoop 08946 [34;1mINF[0m Listening for 22
seconds
<6>17:50:00.062 CbrsDaemon.cpp parseTree 08946 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>17:50:00.064 Sas.cpp post 08946 [36;1m
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
```

```
    "cbsdInfo": {
      "firmwareVersion": "v2.0.5",
      "hardwareVersion": "v1.0.45",
      "model": "CPRI_DEVICE-XXX",
      "softwareVersion": "v1.2.1",
      "vendor": "JMA Wireless"
    },
    "cbsdSerialNumber": "1012482003",
    "fccId": "XM2-X19AX35M2",
    "installationParam": {
      "antennaAzimuth": 70,
      "antennaBeamwidth": 45,
      "antennaDowntilt": 36,
      "antennaGain": 0,
      "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
      "eirpCapability": 15,
      "height": 20.0,
      "heightType": "AMSL",
      "horizontalAccuracy": 49,
      "indoorDeployment": true,
      "latitude": 41.0,
      "longitude": -76.15,
      "verticalAccuracy": 2
    },
    "measCapability": [
      "RECEIVED_POWER_WITH_GRANT"
    ],
    "userId": "abc"
   }
  ]
}
<7>17:50:00.065 Sas.cpp          post          08946 [36;1mDBG[0m {
  "registrationResponse": [
    {
      "response": {
        "responseCode": 200
      }
    }
  ]
}
```

9.11 Log file for test case ID: WINNF.FT.C.GRA.1


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>18:00:35.630 Sas.cpp post 09634 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>18:00:35.630 Sas.cpp post 09634 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>18:00:35.674 Sas.cpp post 09634 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 400
   }
  }
 ]
}
<3>18:00:35.674 Grant.cpp grant_main_ 09634 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>18:00:35.674 Grant.cpp grant_main_ 09634 [36;1mDBG[0m ERROR state reset to IDLE
<6>18:00:35.675 CbrsDaemon.cpp onLoop 09634 [34;1mINF[0m Listening for 22 seconds
<7>18:00:35.675 SpvLaunchdProxy.cpp create 09634 [36;1mDBG[0m Added match-rule: "sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>18:00:35.675 SpvLaunchdProxy.cpp create 09634 [36;1mDBG[0m Added match-rule: "sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>18:00:35.676 SpvLaunchdProxy.cpp initSpvLaunchdProxy 09634 [36;1mDBG[0m SpvLaunchd is running.
<7>18:00:35.676 SpvLaunchdProxy.cpp logDBusMessage 09634 [36;1mDBG[0m handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.59 serial=2 path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>18:00:35.676 SpvLaunchdProxy.cpp dbusHandler 09634 [36;1mDBG[0m NameAcquired: :1.59
<7>18:00:35.676 SpvLaunchdProxy.cpp dbusHandler 09634 [36;1mDBG[0m Connection name: :1.59
<7>18:00:35.676 SpvLaunchdProxy.cpp proc_processes 09634 [36;1mDBG[0m JSON processes_el: [{"enbs":[{"cells":[{"cell_id":0,"cell_key":1,"locked":false}],"com_addr":"127.100.1.1","enb_id":54321,"enb_key":1,"enb_name":"CBRS.TDD.QA"}],"pid":9136,"state":"CONNECTED"}]
<7>18:00:35.676 SpvLaunchdProxy.cpp logActiveEnbs 09634 [36;1mDBG[0m Dump activeEnbs_ map: {"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"invalid_cfg":"","state":"CONNECTED"}]}
<6>18:00:36.698 CbrsDaemon.cpp parseTree 09634 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:00:36.701 Sas.cpp post 09634 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>18:00:36.702 Sas.cpp post 09634 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 400
   }
  }
 ]
}
```

9.12 Log file for test case ID: WINNF.FT.C.GRA.2


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>17:54:49.507 Sas.cpp          post            09312 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>17:54:49.507 Sas.cpp          post            09312 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>17:54:49.551 Sas.cpp          post            09312 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 401
   }
  }
 ]
}
<3>17:54:49.551 Grant.cpp        grant_main_     09312 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>17:54:49.551 Grant.cpp        grant_main_     09312 [36;1mDBG[0m ERROR state reset to IDLE
<6>17:54:49.552 CbrsDaemon.cpp   onLoop          09312 [34;1mINF[0m Listening for 22 seconds
<7>17:54:49.553 SpvLaunchdProxy.cpp create       09312 [36;1mDBG[0m Added match-rule: "sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>17:54:49.553 SpvLaunchdProxy.cpp create       09312 [36;1mDBG[0m Added match-rule: "sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>17:54:49.553 SpvLaunchdProxy.cpp initSpvLaunchdProxy 09312 [36;1mDBG[0m SpvLaunchd is running.
<7>17:54:49.553 SpvLaunchdProxy.cpp logDBusMessage   09312 [36;1mDBG[0m handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.56 serial=2 path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>17:54:49.553 SpvLaunchdProxy.cpp dbusHandler  09312 [36;1mDBG[0m NameAcquired: :1.56
<7>17:54:49.553 SpvLaunchdProxy.cpp dbusHandler  09312 [36;1mDBG[0m Connection name: :1.56
<7>17:54:49.553 SpvLaunchdProxy.cpp proc_processes 09312 [36;1mDBG[0m JSON processes_el: [{"enbs":[{"cells":[{"cell_id":0,"cell_key":1,"locked":false}],"com_addr":"127.100.1.1","enb_id":54321,"enb_key":1,"enb_name":"CBRS.TDD.QA"}],"pid":9136,"state":"CONNECTED"}]
<7>17:54:49.553 SpvLaunchdProxy.cpp logActiveEnbs 09312 [36;1mDBG[0m Dump activeEnbs_ map: {"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"invalid_cfg":"","state":"CONNECTED"}]}
<6>17:54:50.575 CbrsDaemon.cpp   parseTree       09312 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>17:54:50.578 Sas.cpp          post            09312 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>17:54:50.579 Sas.cpp          post            09312 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 400
   }
  }
 ]
}
<3>17:54:50.579 Grant.cpp        grant_main_     09312 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>17:54:50.579 Grant.cpp        grant_main_     09312 [36;1mDBG[0m ERROR state reset to IDLE
<6>17:54:50.579 CbrsDaemon.cpp   onLoop          09312 [34;1mINF[0m Listening for 22 seconds
<6>17:55:13.623 CbrsDaemon.cpp   parseTree       09312 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>17:55:13.626 Sas.cpp          post            09312 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>17:55:13.627 Sas.cpp          post            09312 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 400
   }
  }
 ]
}
<3>17:55:13.627 Grant.cpp        grant_main_     09312 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>17:55:13.627 Grant.cpp        grant_main_     09312 [36;1mDBG[0m ERROR state reset to IDLE
```

<6>17:55:13.628 CbrsDaemon.cpp onLoop 09312 [34;1mINF [0m Listening for 22 seconds
<6>17:55:36.672 CbrsDaemon.cpp parseTree 09312 [34;1mINF [0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>17:55:36.675 Sas.cpp post 09312 [36;1mDBG [0m {
"grantRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"operationParam": {
"maxEirp": 0,
"operationFrequencyRange": {
"highFrequency": 3630000000,
"lowFrequency": 3620000000
}
}
}
]
}
<7>17:55:36.676 Sas.cpp post 09312 [36;1mDBG [0m {
"grantResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"response": {
"responseCode": 400
}
}
]
}

9.13 Log file for test case ID: WINNF.FT.C.HBT.1


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>18:03:47.512 Sas.cpp post 10103 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>18:03:47.512 Sas.cpp post 10103 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>18:03:47.555 Sas.cpp post 10103 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T16:03:47Z",
   "grantId": "196342041",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}

<7>18:03:47.555 Sas.cpp post 10103 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "196342041",
   "operationState": "GRANTED"
  }
 ]
}
<7>18:03:47.598 Sas.cpp post 10103 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "196342041",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:07:07Z"
  }
 ]
}
<7>18:03:47.599 SpvLaunchdProxy.cpp create 10103 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>18:03:47.599 SpvLaunchdProxy.cpp create 10103 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>18:03:47.599 SpvLaunchdProxy.cpp initSpvLaunchdProxy 10103 [36;1mDBG[0m SpvLaunchd is running.
<7>18:03:47.600 SpvLaunchdProxy.cpp logDBusMessage 10103 [36;1mDBG[0m handleRequest: signal
sender=org.freedesktop.DBus -> dest=:1.65 serial=2 path=/org/freedesktop/DBus; interface=org.freedesktop.DBus;
member=NameAcquired; signature=s
<7>18:03:47.600 SpvLaunchdProxy.cpp dbusHandler 10103 [36;1mDBG[0m NameAcquired: :1.65
<7>18:03:47.600 SpvLaunchdProxy.cpp dbusHandler 10103 [36;1mDBG[0m Connection name: :1.65
<7>18:03:47.600 SpvLaunchdProxy.cpp proc_processes 10103 [36;1mDBG[0m JSON processes_el:
[{"enbs":[{"cells":[{"cell_id":0,"cell_key":1,"locked":false}],"com_addr":"127.100.1.1","enb_id":54321,"enb_key":1,
"enb_name":"CBRS.TDD.QA"}],"pid":9968,"state":"CONNECTED"}]
<7>18:03:47.600 SpvLaunchdProxy.cpp logActiveEnbs 10103 [36;1mDBG[0m Dump activeEnbs_ map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"invalid_cfg":"",
"state":"CONNECTED"}]}
<6>18:03:47.600 ManagerEnb.cpp command 10103 [34;1mINF[0m Sending tx_expire to eNB(1), with
expiration: 60000
<6>18:03:47.600 CbrsDaemon.cpp onLoop 10103 [34;1mINF[0m Listening for 22 seconds
<6>18:03:47.739 Enb.cpp onData 10105 [34;1mINF[0m Answer received from eNB (1): flags(129),
{"message":"tx_expire"}
<6>18:04:10.644 CbrsDaemon.cpp parseTree 10103 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl
55990
<7>18:04:10.647 Sas.cpp post 10103 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "196342041",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>18:04:10.651 Sas.cpp post 10103 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "196342041",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:07:30Z"
  }
 ]
}
<6>18:04:10.651 ManagerEnb.cpp command 10103 [34;1mINF[0m Sending tx_expire to eNB(1), with
expiration: 60000
<6>18:04:10.652 CbrsDaemon.cpp onLoop 10103 [34;1mINF[0m Listening for 22 seconds
<6>18:04:10.752 Enb.cpp onData 10105 [34;1mINF[0m Answer received from eNB (1): flags(129),
{"message":"tx_expire"}
<6>18:04:33.695 CbrsDaemon.cpp parseTree 10103 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl
55990
<7>18:04:33.698 Sas.cpp post 10103 [36;1mDBG[0m {
```

```
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "196342041",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>18:04:33.702 Sas.cpp          post            10103 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "196342041",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:07:53Z"
  }
 ]
}
<6>18:04:33.702 ManagerEnb.cpp     command          10103 [34;1mINF[0m Sending tx_expire
to eNB(1), with expiration: 60000
<6>18:04:33.703 CbrsDaemon.cpp     onLoop           10103 [34;1mINF[0m Listening for 22
seconds
<6>18:04:33.803 Enb.cpp          onData           10105 [34;1mINF[0m Answer received from
eNB (1): flags(129), {"message":"tx_expire"}
<6>18:04:56.750 CbrsDaemon.cpp     parseTree        10103 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>18:04:56.753 Sas.cpp          post            10103 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "196342041",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>18:04:56.758 Sas.cpp          post            10103 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "196342041",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:08:16Z"
  }
 ]
}
<6>18:04:56.758 ManagerEnb.cpp     command          10103 [34;1mINF[0m Sending tx_expire
to eNB(1), with expiration: 60000
<6>18:04:56.758 CbrsDaemon.cpp     onLoop           10103 [34;1mINF[0m Listening for 22
seconds
<6>18:04:56.858 Enb.cpp          onData           10105 [34;1mINF[0m Answer received from
eNB (1): flags(129), {"message":"tx_expire"}
<6>18:05:19.806 CbrsDaemon.cpp     parseTree        10103 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>18:05:19.809 Sas.cpp          post            10103 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "196342041",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>18:05:19.813 Sas.cpp          post            10103 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "196342041",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:08:39Z"
  }
 ]
}
<6>18:05:19.813 ManagerEnb.cpp     command          10103 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>18:05:19.814 CbrsDaemon.cpp     onLoop           10103 [34;1mINF[0m Listening for 22 seconds
<6>18:05:19.914 Enb.cpp          onData           10105 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>18:05:42.856 CbrsDaemon.cpp     parseTree        10103 [34;1mINF[0m Found CBRS Cell: cell_id 0,
earfcn_dl 55990
<7>18:05:42.859 Sas.cpp          post            10103 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "196342041",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>18:05:42.863 Sas.cpp          post            10103 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "196342041",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:09:02Z"
  }
 ]
}
```

9.14 Log file for test case ID: WINNF.FT.C.HBT.3


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>18:08:34.754 Sas.cpp post 10286 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>18:08:34.754 Sas.cpp post 10286 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>18:08:34.797 Sas.cpp post 10286 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T16:08:34Z",
   "grantId": "537569458",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>18:08:34.797 Sas.cpp post 10286 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "537569458",
   "operationState": "GRANTED"
  }
 ]
}
<7>18:08:34.840 Sas.cpp post 10286 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "537569458",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:11:54Z"
  }
 ]
}
<7>18:08:34.841 SpvLaunchdProxy.cpp create 10286 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>18:08:34.841 SpvLaunchdProxy.cpp create 10286 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>18:08:34.841 SpvLaunchdProxy.cpp initSpvLaunchdProxy 10286 [36;1mDBG[0m SpvLaunchd is running.
<7>18:08:34.841 SpvLaunchdProxy.cpp logDBusMessage 10286 [36;1mDBG[0m handleRequest: signal
sender=org.freedesktop.DBus -> dest=:1.67 serial=2 path=/org/freedesktop/DBus;
interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>18:08:34.841 SpvLaunchdProxy.cpp dbusHandler 10286 [36;1mDBG[0m NameAcquired: :1.67
<7>18:08:34.841 SpvLaunchdProxy.cpp dbusHandler 10286 [36;1mDBG[0m Connection name: :1.67
<7>18:08:34.841 SpvLaunchdProxy.cpp proc_processes 10286 [36;1mDBG[0m JSON processes_el:
[{"enbs":[{"cells":[{"cell_id":0,"cell_key":1,"locked":false}],"com_addr":"127.100.1.1","enb_id":54321,"enb_key":
1,"enb_name":"CBRS.TDD.QA"}],"pid":9968,"state":"CONNECTED"}]
<7>18:08:34.841 SpvLaunchdProxy.cpp logActiveEnbs 10286 [36;1mDBG[0m Dump activeEnbs_ map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"invalid_cfg":"
","state":"CONNECTED"}]}
<6>18:08:34.841 ManagerEnb.cpp command 10286 [34;1mINF[0m Sending tx_expire to eNB(1), with
expiration: 60000
<6>18:08:34.842 CbrsDaemon.cpp onLoop 10286 [34;1mINF[0m Listening for 22 seconds
<6>18:08:34.981 Enb.cpp onData 10288 [34;1mINF[0m Answer received from eNB (1): flags(129),
{"message":"tx_expire"}
<6>18:08:57.886 CbrsDaemon.cpp parseTree 10286 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl
55990
<7>18:08:57.889 Sas.cpp post 10286 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "537569458",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>18:08:57.893 Sas.cpp post 10286 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "537569458",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:12:17Z"
  }
 ]
}
<6>18:08:57.893 ManagerEnb.cpp command 10286 [34;1mINF[0m Sending tx_expire to eNB(1), with
expiration: 60000
<6>18:08:57.894 CbrsDaemon.cpp onLoop 10286 [34;1mINF[0m Listening for 22 seconds
<6>18:08:57.993 Enb.cpp onData 10288 [34;1mINF[0m Answer received from eNB (1): flags(129),
{"message":"tx_expire"}
<6>18:09:20.941 CbrsDaemon.cpp parseTree 10286 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl
55990
<7>18:09:20.944 Sas.cpp post 10286 [36;1mDBG[0m {
```

```
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "537569458",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>18:09:20.947 Sas.cpp          post            10286 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "537569458",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:12:40Z"
  }
 ]
}
<6>18:09:20.948 ManagerEnb.cpp     command          10286 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>18:09:20.948 CbrsDaemon.cpp     onLoop           10286 [34;1mINF[0m Listening for 22 seconds
<6>18:09:21.048 Enb.cpp            onData           10288 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>18:09:43.995 CbrsDaemon.cpp     parseTree        10286 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:09:43.998 Sas.cpp          post            10286 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "537569458",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>18:09:44.003 Sas.cpp          post            10286 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "537569458",
   "response": {
    "responseCode": 105
   },
   "transmitExpireTime": "2019-05-27T16:09:44Z"
  }
 ]
}
```

9.15 Log file for test case ID: WINNF.FT.C.HBT.4


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>18:13:16.371 Sas.cpp post 10541 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>18:13:16.371 Sas.cpp post 10541 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>18:13:16.415 Sas.cpp post 10541 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T16:13:16Z",
   "grantId": "931008927",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>18:13:16.415 Sas.cpp post 10541 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "931008927",
   "operationState": "GRANTED"
  }
 ]
}
<7>18:13:16.457 Sas.cpp post 10541 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "931008927",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:16:36Z"
  }
 ]
}
<7>18:13:16.458 SpvLaunchdProxy.cpp create 10541 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>18:13:16.458 SpvLaunchdProxy.cpp create 10541 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>18:13:16.458 SpvLaunchdProxy.cpp initSpvLaunchdProxy 10541 [36;1mDBG[0m SpvLaunchd is running.
<7>18:13:16.458 SpvLaunchdProxy.cpp logDBusMessage 10541 [36;1mDBG[0m handleRequest: signal
sender=org.freedesktop.DBus -> dest=:1.70 serial=2 path=/org/freedesktop/DBus;
interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>18:13:16.458 SpvLaunchdProxy.cpp dbusHandler 10541 [36;1mDBG[0m NameAcquired: :1.70
<7>18:13:16.458 SpvLaunchdProxy.cpp dbusHandler 10541 [36;1mDBG[0m Connection name: :1.70
<7>18:13:16.458 SpvLaunchdProxy.cpp proc_processes 10541 [36;1mDBG[0m JSON processes_el:
[{"enbs":[{"cells":[{"cell_id":0,"cell_key":1,"locked":false}],"com_addr":"127.100.1.1","enb_id":54321,"enb_key":
1,"enb_name":"CBRS.TDD.QA"}],"pid":10367,"state":"CONNECTED"}]
<7>18:13:16.458 SpvLaunchdProxy.cpp logActiveEnbs 10541 [36;1mDBG[0m Dump activeEnbs_ map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"invalid_cfg":"
","state":"CONNECTED"}]}
<6>18:13:16.458 ManagerEnb.cpp command 10541 [34;1mINF[0m Sending tx_expire to eNB(1), with
expiration: 60000
<6>18:13:16.459 CbrsDaemon.cpp onLoop 10541 [34;1mINF[0m Listening for 2 seconds
<6>18:13:16.598 Enb.cpp onData 10543 [34;1mINF[0m Answer received from eNB (1): flags(129),
{"message":"tx_expire"}
<6>18:13:19.483 CbrsDaemon.cpp parseTree 10541 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl
55990
<7>18:13:19.485 Sas.cpp post 10541 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "931008927",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>18:13:19.488 Sas.cpp post 10541 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "931008927",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:16:39Z"
  }
 ]
}
<6>18:13:19.488 ManagerEnb.cpp command 10541 [34;1mINF[0m Sending tx_expire to eNB(1), with
expiration: 60000
<6>18:13:19.489 CbrsDaemon.cpp onLoop 10541 [34;1mINF[0m Listening for 2 seconds
<6>18:13:19.589 Enb.cpp onData 10543 [34;1mINF[0m Answer received from eNB (1): flags(129),
{"message":"tx_expire"}
<6>18:13:22.512 CbrsDaemon.cpp parseTree 10541 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl
55990
<7>18:13:22.515 Sas.cpp post 10541 [36;1mDBG[0m {
```

```
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "931008927",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>18:13:22.518 Sas.cpp post 10541 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "931008927",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:16:42Z"
  }
 ]
}
<6>18:13:22.518 ManagerEnb.cpp command 10541 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>18:13:22.519 CbrsDaemon.cpp onLoop 10541 [34;1mINF[0m Listening for 2 seconds
<6>18:13:22.619 Enb.cpp onData 10543 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>18:13:25.543 CbrsDaemon.cpp parseTree 10541 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:13:25.545 Sas.cpp post 10541 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "931008927",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>18:13:25.548 Sas.cpp post 10541 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "931008927",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:16:45Z"
  }
 ]
}
<6>18:13:25.548 ManagerEnb.cpp command 10541 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>18:13:25.549 CbrsDaemon.cpp onLoop 10541 [34;1mINF[0m Listening for 2 seconds
<6>18:13:25.649 Enb.cpp onData 10543 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>18:13:28.573 CbrsDaemon.cpp parseTree 10541 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:13:28.575 Sas.cpp post 10541 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "931008927",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>18:13:28.579 Sas.cpp post 10541 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "931008927",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:16:48Z"
  }
 ]
}
<6>18:13:28.579 ManagerEnb.cpp command 10541 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>18:13:28.580 CbrsDaemon.cpp onLoop 10541 [34;1mINF[0m Listening for 2 seconds
<6>18:13:28.679 Enb.cpp onData 10543 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>18:13:31.601 CbrsDaemon.cpp parseTree 10541 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:13:31.604 Sas.cpp post 10541 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "931008927",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>18:13:31.608 Sas.cpp post 10541 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "931008927",
   "response": {
    "responseCode": 500
   },
   "transmitExpireTime": "2019-05-27T16:13:31Z"
  }
 ]
}
<3>18:13:31.608 Grant.cpp grant_main_ 10541 [31;1mERR[0m Heartbeat procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, grantId 931008927
<7>18:13:31.608 Grant.cpp grant_main_ 10541 [36;1mDBG[0m ERROR state reset to IDLE
<6>18:13:31.608 ManagerEnb.cpp command 10541 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 0
<6>18:13:31.608 CbrsDaemon.cpp onLoop 10541 [34;1mINF[0m Listening for 2 seconds
<6>18:13:31.708 Enb.cpp onData 10543 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>18:13:34.632 CbrsDaemon.cpp parseTree 10541 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:13:34.635 Sas.cpp post 10541 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>18:13:34.636 Sas.cpp post 10541 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 400
   }
  }
 ]
}
<3>18:13:34.636 Grant.cpp grant_main_ 10541 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>18:13:34.636 Grant.cpp grant_main_ 10541 [36;1mDBG[0m ERROR state reset to IDLE
<6>18:13:34.636 CbrsDaemon.cpp onLoop 10541 [34;1mINF[0m Listening for 2 seconds
<6>18:13:37.660 CbrsDaemon.cpp parseTree 10541 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:13:37.662 Sas.cpp post 10541 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
```

<7>18:13:37.663 Sas.cpp post 10541 [36;1mDBG[0m {
"grantResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"response": {
"responseCode": 400
}
}
]
}
<3>18:13:37.663 Grant.cpp grant_main_ 10541 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>18:13:37.663 Grant.cpp grant_main_ 10541 [36;1mDBG[0m ERROR state reset to IDLE
<6>18:13:37.664 CbrsDaemon.cpp onLoop 10541 [34;1mINF[0m Listening for 2 seconds
<6>18:13:40.688 CbrsDaemon.cpp parseTree 10541 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:13:40.690 Sas.cpp post 10541 [36;1mDBG[0m {
"grantRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"operationParam": {
"maxEirp": 0,
"operationFrequencyRange": {
"highFrequency": 3630000000,
"lowFrequency": 3620000000
}
}
}
]
}
<7>18:13:40.691 Sas.cpp post 10541 [36;1mDBG[0m {
"grantResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"response": {
"responseCode": 400
}
}
]
}
<3>18:13:40.691 Grant.cpp grant_main_ 10541 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>18:13:40.691 Grant.cpp grant_main_ 10541 [36;1mDBG[0m ERROR state reset to IDLE
<6>18:13:40.692 CbrsDaemon.cpp onLoop 10541 [34;1mINF[0m Listening for 2 seconds
<6>18:13:43.719 CbrsDaemon.cpp parseTree 10541 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:13:43.721 Sas.cpp post 10541 [36;1mDBG[0m {
"grantRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"operationParam": {
"maxEirp": 0,
"operationFrequencyRange": {
"highFrequency": 3630000000,
"lowFrequency": 3620000000
}
}
}
]
}
<7>18:13:43.722 Sas.cpp post 10541 [36;1mDBG[0m {
"grantResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"response": {
"responseCode": 400
}
}
]
}
<3>18:13:43.722 Grant.cpp grant_main_ 10541 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>18:13:43.722 Grant.cpp grant_main_ 10541 [36;1mDBG[0m ERROR state reset to IDLE
<6>18:13:43.723 CbrsDaemon.cpp onLoop 10541 [34;1mINF[0m Listening for 2 seconds
<6>18:13:46.746 CbrsDaemon.cpp parseTree 10541 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:13:46.749 Sas.cpp post 10541 [36;1mDBG[0m {
"grantRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"operationParam": {
"maxEirp": 0,
"operationFrequencyRange": {
"highFrequency": 3630000000,
"lowFrequency": 3620000000
}
}
}
]
}
<7>18:13:46.750 Sas.cpp post 10541 [36;1mDBG[0m {
"grantResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"response": {
"responseCode": 400
}
}
]
}

9.16 Log file for test case ID: WINNF.FT.C.HBT.5


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>18:17:15.510 Sas.cpp          post            11174 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>18:17:15.510 Sas.cpp          post            11174 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
<7>18:17:15.554 Sas.cpp          post            11174 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T16:17:15Z",
   "grantId": "755676189",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
}
<7>18:17:15.554 Sas.cpp          post            11174 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "755676189",
   "operationState": "GRANTED"
  }
 ]
}
<7>18:17:15.597 Sas.cpp          post            11174 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "755676189",
   "response": {
    "responseCode": 501
   },
   "transmitExpireTime": "2019-05-27T16:17:15Z"
  }
 ]
}
<3>18:17:15.597 Grant.cpp        grant_main_     11174 [31;1mERR[0m Heartbeat procedure failed
for CBSD XM2-X19AX35M2Mock-SAS1012482003, grantId 755676189
<7>18:17:15.597 Grant.cpp        grant_main_     11174 [36;1mDBG[0m ERROR state reset to IDLE
<6>18:17:15.598 CbrsDaemon.cpp   onLoop          11174 [34;1mINF[0m Listening for 2 seconds
<7>18:17:15.599 SpvLaunchdProxy.cpp  create      11174 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>18:17:15.599 SpvLaunchdProxy.cpp  create      11174 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>18:17:15.599 SpvLaunchdProxy.cpp  initSpvLaunchdProxy  11174 [36;1mDBG[0m SpvLaunchd is
running.
<7>18:17:15.599 SpvLaunchdProxy.cpp  logDBusMessage  11174 [36;1mDBG[0m handleRequest:
signal sender=org.freedesktop.DBus -> dest=:1.78 serial=2 path=/org/freedesktop/DBus;
interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>18:17:15.599 SpvLaunchdProxy.cpp  dbusHandler  11174 [36;1mDBG[0m NameAcquired:
:1.78
<7>18:17:15.599 SpvLaunchdProxy.cpp  dbusHandler  11174 [36;1mDBG[0m Connection name:
:1.78
<7>18:17:15.599 SpvLaunchdProxy.cpp  proc_processes  11174 [36;1mDBG[0m JSON processes_el:
[{"enbs":[{"cells":[{"cell_id":0,"cell_key":1,"locked":false}],"com_addr":"127.100.1.1","enb_id":54321,
"enb_key":1,"enb_name":"CBRS.TDD.QA"}],"pid":11157,"state":"CONNECTED"}]
<7>18:17:15.599 SpvLaunchdProxy.cpp  logActiveEnbs  11174 [36;1mDBG[0m Dump activeEnbs_
map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"in
valid_cfg":"","state":"CONNECTED"}]}
<6>18:17:16.621 CbrsDaemon.cpp   parseTree       11174 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>18:17:16.624 Sas.cpp          post            11174 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
```

<7>18:17:16.872 Sas.cpp post 11174 [36;1mDBG[0m [
"Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2 Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+the+node+%3A+grantRequest+not+exists+in+the+expected+json+file+%3AHBT_Post_HB_Error_501_granted.json. If not click the link.",
"Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2 Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+error+accoured+in+the+last+request+from+the+CBRS. If not click the link.",
"Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2 Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+error+accoured+in+the+last+request+from+the+CBRS. If not click the link."
]
<7>18:17:16.872 Sas.cpp post 11174 [36;1mDBG[0m null
<3>18:17:16.872 Grant.cpp grant_main_ 11174 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<3>18:17:16.872 Grant.cpp grant_main_ 11174 [31;1mERR[0m Grant error detail: Unable to parse: <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>Redirecting...</title>
<h1>Redirecting...</h1>
<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+the+node+%3A+grantRequest+not+exists+in+the+expected+json+file+%3AHBT_Post_HB_Error_501_granted.json. If not click the link.
<3>18:17:16.872 Grant.cpp grant_main_ 11174 [31;1mERR[0m Grant error detail: Unable to parse: <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>Redirecting...</title>
<h1>Redirecting...</h1>
<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+error+accoured+in+the+last+request+from+the+CBRS. If not click the link.
<3>18:17:16.872 Grant.cpp grant_main_ 11174 [31;1mERR[0m Grant error detail: Unable to parse: <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>Redirecting...</title>
<h1>Redirecting...</h1>
<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+error+accoured+in+the+last+request+from+the+CBRS. If not click the link.
<7>18:17:16.872 Grant.cpp grant_main_ 11174 [36;1mDBG[0m ERROR state reset to IDLE
<6>18:17:16.873 CbrsDaemon.cpp onLoop 11174 [34;1mINF[0m Listening for 2 seconds
<6>18:17:19.896 CbrsDaemon.cpp parseTree 11174 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:17:19.899 Sas.cpp post 11174 [36;1mDBG[0m {
"grantRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"operationParam": {
"maxEirp": 0,
"operationFrequencyRange": {
"highFrequency": 3630000000,
"lowFrequency": 3620000000
}
}
}
]
}

<7>18:17:20.149 Sas.cpp post 11174 [36;1mDBG[0m [
"Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2 Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+error+accoured+in+the+last+request+from+the+CBRS. If not click the link.",
"Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2 Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+error+accoured+in+the+last+request+from+the+CBRS. If not click the link.",
"Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2 Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+error+accoured+in+the+last+request+from+the+CBRS. If not click the link."
]

9.17 Log file for test case ID: WINNF.FT.C.HBT.6


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>18:20:41.229 Sas.cpp          post            11540 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
<6>18:20:41.230 CbrsDaemon.cpp      onLoop          11540 [34;1mINF[0m Listening for 2 seconds
<7>18:20:41.231 SpvLaunchdProxy.cpp  create          11540 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>18:20:41.231 SpvLaunchdProxy.cpp  create          11540 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>18:20:41.231 SpvLaunchdProxy.cpp  initSpvLaunchdProxy  11540 [36;1mDBG[0m SpvLaunchd
is running.
<7>18:20:41.231 SpvLaunchdProxy.cpp  logDBusMessage      11540 [36;1mDBG[0m
handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.83 serial=2
path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired;
signature=s
<7>18:20:41.231 SpvLaunchdProxy.cpp  dbusHandler       11540 [36;1mDBG[0m NameAcquired:
:1.83
<7>18:20:41.231 SpvLaunchdProxy.cpp  dbusHandler       11540 [36;1mDBG[0m Connection
name: :1.83
<6>18:20:42.254 CbrsDaemon.cpp      parseTree        11540 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>18:20:42.256 CbrsDaemon.cpp      onLoop          11540 [34;1mINF[0m Listening for 2 seconds
<6>18:20:45.280 CbrsDaemon.cpp      parseTree        11540 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>18:20:45.283 CbrsDaemon.cpp      persistEntities    11540 [36;1mDBG[0m Grant for cell 0,
belonging to eNB 1 created.
<7>18:20:45.284 Sas.cpp          post            11540 [36;1mDBG[0m {
}
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>18:20:45.288 Sas.cpp          post            11540 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T16:20:45Z",
   "grantId": "167155198",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>18:20:45.288 Sas.cpp          post            11540 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "167155198",
   "operationState": "GRANTED"
  }
 ]
}
<7>18:20:45.331 Sas.cpp          post            11540 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "167155198",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T16:24:05Z"
  }
 ]
}
<6>18:20:45.332 CbrsDaemon.cpp      onLoop          11540 [34;1mINF[0m Listening for 2 seconds
<7>18:20:45.332 SpvLaunchdProxy.cpp  logDBusMessage      11540 [36;1mDBG[0m handleRequest:
signal sender=:1.0 -> dest=(null) serial=113 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StartProcess; signature=s
<7>18:20:45.332 SpvLaunchdProxy.cpp  logActiveEnbs       11540 [36;1mDBG[0m Dump activeEnbs_
map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"in
valid_cfg":"","state":"CONNECTED"}]}
<6>18:20:46.354 CbrsDaemon.cpp      parseTree        11540 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>18:20:46.356 Sas.cpp          post            11540 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "167155198",
   "operationState": "AUTHORIZED"
  }
 ]
}
```

<7>18:20:46.359 Sas.cpp post 11540 [36;1mDBG[0m {
"heartbeatResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "167155198",
"response": {
"responseCode": 0
},
"transmitExpireTime": "2019-05-27T16:24:06Z"
}
]
}
<6>18:20:46.359 ManagerEnb.cpp command 11540 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>18:20:46.360 CbrsDaemon.cpp onLoop 11540 [34;1mINF[0m Listening for 2 seconds
<6>18:20:46.499 Enb.cpp onData 11571 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>18:20:49.384 CbrsDaemon.cpp parseTree 11540 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:20:49.387 Sas.cpp post 11540 [36;1mDBG[0m {
"heartbeatRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "167155198",
"operationState": "AUTHORIZED"
}
]
}
<7>18:20:49.390 Sas.cpp post 11540 [36;1mDBG[0m {
"heartbeatResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "167155198",
"response": {
"responseCode": 0
},
"transmitExpireTime": "2019-05-27T16:24:09Z"
}
]
}
<6>18:20:49.390 ManagerEnb.cpp command 11540 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>18:20:49.391 CbrsDaemon.cpp onLoop 11540 [34;1mINF[0m Listening for 2 seconds
<6>18:20:49.491 Enb.cpp onData 11571 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>18:20:52.415 CbrsDaemon.cpp parseTree 11540 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:20:52.418 Sas.cpp post 11540 [36;1mDBG[0m {
"heartbeatRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "167155198",
"operationState": "AUTHORIZED"
}
]
}
<7>18:20:52.421 Sas.cpp post 11540 [36;1mDBG[0m {
"heartbeatResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "167155198",
"response": {
"responseCode": 501
},
"transmitExpireTime": "2019-05-27T16:20:52Z"
}
]
}
<3>18:20:52.421 Grant.cpp grant_main_ 11540 [31;1mERR[0m Heartbeat procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, grantId 167155198
<7>18:20:52.421 Grant.cpp grant_main_ 11540 [36;1mDBG[0m ERROR state reset to IDLE
<6>18:20:52.421 ManagerEnb.cpp command 11540 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 0
<6>18:20:52.422 CbrsDaemon.cpp onLoop 11540 [34;1mINF[0m Listening for 2 seconds
<6>18:20:52.522 Enb.cpp onData 11571 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>18:20:55.446 CbrsDaemon.cpp parseTree 11540 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>18:20:55.449 Sas.cpp post 11540 [36;1mDBG[0m {
"grantRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"operationParam": {
"maxEirp": 0,
"operationFrequencyRange": {
"highFrequency": 3630000000,
"lowFrequency": 3620000000
}
}
}
]
}
<7>18:20:55.700 Sas.cpp post 11540 [36;1mDBG[0m [
"Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2 Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+the+node+%3A+grantRequest+not+exists+in+the+expected+json+file+%3AHBT_Post_HB_Error_501.json. If not click the link.",
"Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2 Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+error+accoured+in+the+last+request+from+the+CBRS. If not click the link.",
"Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2 Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+error+accoured+in+the+last+request+from+the+CBRS. If not click the link."
]
<7>18:20:55.700 Sas.cpp post 11540 [36;1mDBG[0m null
<3>18:20:55.700 Grant.cpp grant_main_ 11540 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<3>18:20:55.700 Grant.cpp grant_main_ 11540 [31;1mERR[0m Grant error detail: Unable to parse: <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>Redirecting...</title>
<h1>Redirecting...</h1>
<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+the+node+%3A+grantRequest+not+exists+in+the+expected+json+file+%3AHBT_Post_HB_Error_501.json. If not click the link.
<3>18:20:55.700 Grant.cpp grant_main_ 11540 [31;1mERR[0m Grant error detail: Unable to parse: <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>Redirecting...</title>
<h1>Redirecting...</h1>
<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+error+accoured+in+the+last+request+from+the+CBRS. If not click the link.
<3>18:20:55.700 Grant.cpp grant_main_ 11540 [31;1mERR[0m Grant error detail: Unable to parse: <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>Redirecting...</title>
<h1>Redirecting...</h1>

```
<p>You should be redirected automatically to target URL: <a
href="/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS">/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS</a>.  If not click the link.
<7>18:20:55.700 Grant.cpp          grant_main_        11540 [36;1mDBG[0m ERROR state reset to
IDLE
<6>18:20:55.700 CbrsDaemon.cpp      onLoop             11540 [34;1mINF[0m Listening for 2 seconds
<6>18:20:58.724 CbrsDaemon.cpp      parseTree          11540 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>18:20:58.727 Sas.cpp            post               11540 [36;1mDBG[0m {
  "grantRequest": [
    {
      "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
      "operationParam": {
        "maxEirp": 0,
        "operationFrequencyRange": {
          "highFrequency": 3630000000,
          "lowFrequency": 3620000000
        }
      }
    }
  ]
}
<7>18:20:58.977 Sas.cpp            post               11540 [36;1mDBG[0m [
  "Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2
Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be
redirected automatically to target URL: <a href=\"/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS\">/shutdown?validationMessage=ERROR+
-+error+accoured+in+the+last+request+from+the+CBRS</a>.  If not click the link.",
  "Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2
Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be
redirected automatically to target URL: <a href=\"/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS\">/shutdown?validationMessage=ERROR+
-+error+accoured+in+the+last+request+from+the+CBRS</a>.  If not click the link.",
  "Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2
Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be
redirected automatically to target URL: <a href=\"/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS\">/shutdown?validationMessage=ERROR+
-+error+accoured+in+the+last+request+from+the+CBRS</a>.  If not click the link."
]
<7>18:20:58.977 Sas.cpp            post               11540 [36;1mDBG[0m null
<3>18:20:58.977 Grant.cpp          grant_main_        11540 [31;1mERR[0m Grant procedure failed
for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<3>18:20:58.977 Grant.cpp          grant_main_        11540 [31;1mERR[0m Grant error detail:
Unable to parse: <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>Redirecting...</title>
<h1>Redirecting...</h1>
<p>You should be redirected automatically to target URL: <a
href="/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS">/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS</a>.  If not click the link.
<3>18:20:58.977 Grant.cpp          grant_main_        11540 [31;1mERR[0m Grant error detail:
Unable to parse: <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>Redirecting...</title>
<h1>Redirecting...</h1>
<p>You should be redirected automatically to target URL: <a
href="/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS">/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS</a>.  If not click the link.
<3>18:20:58.977 Grant.cpp          grant_main_        11540 [31;1mERR[0m Grant error detail:
Unable to parse: <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2 Final//EN">
<title>Redirecting...</title>
<h1>Redirecting...</h1>
<p>You should be redirected automatically to target URL: <a
href="/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS">/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS</a>.  If not click the link.
<7>18:20:58.977 Grant.cpp          grant_main_        11540 [36;1mDBG[0m ERROR state reset to
IDLE
<6>18:20:58.977 CbrsDaemon.cpp      onLoop             11540 [34;1mINF[0m Listening for 2 seconds
<6>18:21:02.000 CbrsDaemon.cpp      parseTree          11540 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>18:21:02.003 Sas.cpp            post               11540 [36;1mDBG[0m {
  "grantRequest": [
    {
      "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
      "operationParam": {
        "maxEirp": 0,
        "operationFrequencyRange": {
          "highFrequency": 3630000000,
          "lowFrequency": 3620000000
        }
      }
    }
  ]
}
<6>18:21:02.158 Daemon.cpp         signalTreatment    11540 [34;1mINF[0m Received signal: 2.
<7>18:21:02.213 Sas.cpp            post               11540 [36;1mDBG[0m [
  "Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2
Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected
automatically to target URL: <a href=\"/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS\">/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS</a>.  If not click the link.",
  "Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2
Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected
automatically to target URL: <a href=\"/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS\">/shutdown?validationMessage=ERROR+-
+error+accoured+in+the+last+request+from+the+CBRS</a>.  If not click the link.",
  "Malformed message"
]
```

9.18 Log file for test case ID: WINNF.FT.C.HBT.7


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>19:33:59.978 Sas.cpp post 12756 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>19:33:59.979 CbrsDaemon.cpp onLoop 12756 [34;1mINF[0m Listening for 2 seconds
<7>19:33:59.979 SpvLaunchdProxy.cpp create 12756 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>19:33:59.979 SpvLaunchdProxy.cpp create 12756 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>19:33:59.979 SpvLaunchdProxy.cpp initSpvLaunchdProxy 12756 [36;1mDBG[0m SpvLaunchd
is running.
<7>19:33:59.980 SpvLaunchdProxy.cpp logDBusMessage 12756 [36;1mDBG[0m
handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.85 serial=2
path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired;
signature=s
<7>19:33:59.980 SpvLaunchdProxy.cpp dbusHandler 12756 [36;1mDBG[0m NameAcquired:
:1.85
<7>19:33:59.980 SpvLaunchdProxy.cpp dbusHandler 12756 [36;1mDBG[0m Connection
name: :1.85
<6>19:34:01.002 CbrsDaemon.cpp parseTree 12756 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>19:34:01.004 CbrsDaemon.cpp onLoop 12756 [34;1mINF[0m Listening for 2 seconds
<7>19:34:02.324 SpvLaunchdProxy.cpp logDBusMessage 12756 [36;1mDBG[0m handleRequest:
signal sender=:1.0 -> dest=(null) serial=119 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StartProcess; signature=s
<7>19:34:02.324 SpvLaunchdProxy.cpp logActiveEnbs 12756 [36;1mDBG[0m Dump activeEnbs_
map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"in
valid_cfg":"","state":"CONNECTED"}]}
<6>19:34:03.347 CbrsDaemon.cpp parseTree 12756 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:34:03.349 CbrsDaemon.cpp persistEntities 12756 [36;1mDBG[0m Grant for cell 0,
belonging to eNB 1 created.
<7>19:34:03.350 Sas.cpp post 12756 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>19:34:03.355 Sas.cpp post 12756 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T17:34:03Z",
   "grantId": "649460397",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>19:34:03.355 Sas.cpp post 12756 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "649460397",
   "operationState": "GRANTED"
  }
 ]
}
<7>19:34:03.397 Sas.cpp post 12756 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "649460397",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:37:23Z"
  }
 ]
}
```

<6>19:34:03.397 ManagerEnb.cpp command 12756 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>19:34:03.398 CbrsDaemon.cpp onLoop 12756 [34;1mINF[0m Listening for 2 seconds
<6>19:34:04.046 Enb.cpp onData 12787 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>19:34:06.421 CbrsDaemon.cpp parseTree 12756 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:34:06.424 Sas.cpp post 12756 [36;1mDBG[0m {
"heartbeatRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "649460397",
"operationState": "AUTHORIZED"
}
]
}
<7>19:34:06.427 Sas.cpp post 12756 [36;1mDBG[0m {
"heartbeatResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "649460397",
"response": {
"responseCode": 0
},
"transmitExpireTime": "2019-05-27T17:37:26Z"
}
]
}
<6>19:34:06.427 ManagerEnb.cpp command 12756 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>19:34:06.428 CbrsDaemon.cpp onLoop 12756 [34;1mINF[0m Listening for 2 seconds
<6>19:34:06.528 Enb.cpp onData 12787 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>19:34:09.451 CbrsDaemon.cpp parseTree 12756 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:34:09.455 Sas.cpp post 12756 [36;1mDBG[0m {
"heartbeatRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "649460397",
"operationState": "AUTHORIZED"
}
]
}
<7>19:34:09.458 Sas.cpp post 12756 [36;1mDBG[0m {
"heartbeatResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "649460397",
"response": {
"responseCode": 0
},
"transmitExpireTime": "2019-05-27T17:37:29Z"
}
]
}
<6>19:34:09.458 ManagerEnb.cpp command 12756 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>19:34:09.459 CbrsDaemon.cpp onLoop 12756 [34;1mINF[0m Listening for 2 seconds
<6>19:34:09.559 Enb.cpp onData 12787 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>19:34:12.483 CbrsDaemon.cpp parseTree 12756 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:34:12.486 Sas.cpp post 12756 [36;1mDBG[0m {
"heartbeatRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "649460397",
"operationState": "AUTHORIZED"
}
]
}
<7>19:34:12.489 Sas.cpp post 12756 [36;1mDBG[0m {
"heartbeatResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "649460397",
"response": {
"responseCode": 502
},
"transmitExpireTime": "2019-05-27T17:34:12Z"
}
]
}
<3>19:34:12.489 Grant.cpp grant_main_ 12756 [31;1mERR[0m Heartbeat procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, grantId 649460397
<7>19:34:12.489 Grant.cpp grant_main_ 12756 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:34:12.489 ManagerEnb.cpp command 12756 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 0
<6>19:34:12.490 CbrsDaemon.cpp onLoop 12756 [34;1mINF[0m Listening for 2 seconds
<6>19:34:12.590 Enb.cpp onData 12787 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>19:34:15.517 CbrsDaemon.cpp parseTree 12756 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:34:15.520 Sas.cpp post 12756 [36;1mDBG[0m {
"grantRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"operationParam": {
"maxEirp": 0,
"operationFrequencyRange": {
"highFrequency": 3630000000,
"lowFrequency": 3620000000
}
}
}
]
}
<7>19:34:15.768 Sas.cpp post 12756 [36;1mDBG[0m [
"Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2 Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+the+node+%3A+grantRequest+not+exists+in+the+expected+json+file+%3AHBT_Post_HB_Error_502.json. If not click the link.",
"Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2 Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+error+accoured+in+the+last+request+from+the+CBRS. If not click the link.",
"Unable to parse: <!DOCTYPE HTML PUBLIC \"-//W3C//DTD HTML 3.2 Final//EN\">\n<title>Redirecting...</title>\n<h1>Redirecting...</h1>\n<p>You should be redirected automatically to target URL: /shutdown?validationMessage=ERROR+-+error+accoured+in+the+last+request+from+the+CBRS. If not click the link."
]

9.19 Log file for test case ID: WINNF.FT.C.HBT.9


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>19:35:39.293 Sas.cpp post 12951 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>19:35:39.293 CbrsDaemon.cpp onLoop 12951 [34;1mINF[0m Listening for 2 seconds
<7>19:35:39.294 SpvLaunchdProxy.cpp create 12951 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>19:35:39.294 SpvLaunchdProxy.cpp create 12951 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>19:35:39.294 SpvLaunchdProxy.cpp initSpvLaunchdProxy 12951 [36;1mDBG[0m SpvLaunchd
is running.
<7>19:35:39.294 SpvLaunchdProxy.cpp logDBusMessage 12951 [36;1mDBG[0m
handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.87 serial=2
path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired;
signature=s
<7>19:35:39.294 SpvLaunchdProxy.cpp dbusHandler 12951 [36;1mDBG[0m NameAcquired:
:1.87
<7>19:35:39.294 SpvLaunchdProxy.cpp dbusHandler 12951 [36;1mDBG[0m Connection
name: :1.87
<6>19:35:40.317 CbrsDaemon.cpp parseTree 12951 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>19:35:40.319 CbrsDaemon.cpp onLoop 12951 [34;1mINF[0m Listening for 2 seconds
<6>19:35:43.343 CbrsDaemon.cpp parseTree 12951 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:35:43.346 CbrsDaemon.cpp persistEntities 12951 [36;1mDBG[0m Grant for cell 0,
belonging to eNB 1 created.
<7>19:35:43.347 Sas.cpp post 12951 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>19:35:43.351 Sas.cpp post 12951 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T17:35:43Z",
   "grantId": "279158631",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>19:35:43.351 Sas.cpp post 12951 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "279158631",
   "operationState": "GRANTED"
  }
 ]
}
<7>19:35:49.522 Sas.cpp post 12951 [36;1mDBG[0m [
 "No data received",
 "No data received",
 "No data received"
]
<7>19:35:49.522 Sas.cpp post 12951 [36;1mDBG[0m null
<3>19:35:49.522 Grant.cpp grant_main_ 12951 [31;1mERR[0m Heartbeat procedure failed
for CBSD XM2-X19AX35M2Mock-SAS1012482003, grantId 279158631
<3>19:35:49.522 Grant.cpp grant_main_ 12951 [31;1mERR[0m Heartbeat error detail: No
data received
<3>19:35:49.522 Grant.cpp grant_main_ 12951 [31;1mERR[0m Heartbeat error detail: No
data received
<3>19:35:49.522 Grant.cpp grant_main_ 12951 [31;1mERR[0m Heartbeat error detail: No
data received
<7>19:35:49.522 Grant.cpp grant_main_ 12951 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:35:49.523 CbrsDaemon.cpp onLoop 12951 [34;1mINF[0m Listening for 2 seconds
```

<7>19:35:49.523 SpvLaunchdProxy.cpp logDBusMessage 12951 [36;1mDBG[0m handleRequest: signal sender=:1.0 -> dest=(null) serial=125 path=/com/jmawireless/jsoft/SpvLaunchd; interface=com.jmawireless.jsoft.SpvLaunchd; member=StartProcess; signature=s
<7>19:35:49.523 SpvLaunchdProxy.cpp logActiveEnbs 12951 [36;1mDBG[0m Dump activeEnbs_ map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"invalid_cfg":"","state":"CONNECTED"}]}
<6>19:35:50.545 CbrsDaemon.cpp parseTree 12951 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:35:50.548 Sas.cpp post 12951 [36;1mDBG[0m {
"grantRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"operationParam": {
"maxEirp": 0,
"operationFrequencyRange": {
"highFrequency": 3630000000,
"lowFrequency": 3620000000
}
}
}
]
}
<7>19:35:56.721 Sas.cpp post 12951 [36;1mDBG[0m [
"No data received",
"No data received",
"No data received"
]
<7>19:35:56.721 Sas.cpp post 12951 [36;1mDBG[0m null
<3>19:35:56.721 Grant.cpp grant_main_ 12951 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<3>19:35:56.721 Grant.cpp grant_main_ 12951 [31;1mERR[0m Grant error detail: No data received
<3>19:35:56.721 Grant.cpp grant_main_ 12951 [31;1mERR[0m Grant error detail: No data received
<3>19:35:56.721 Grant.cpp grant_main_ 12951 [31;1mERR[0m Grant error detail: No data received
<7>19:35:56.721 Grant.cpp grant_main_ 12951 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:35:56.722 CbrsDaemon.cpp onLoop 12951 [34;1mINF[0m Listening for 2 seconds
<6>19:35:59.746 CbrsDaemon.cpp parseTree 12951 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:35:59.748 Sas.cpp post 12951 [36;1mDBG[0m {
"grantRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"operationParam": {
"maxEirp": 0,
"operationFrequencyRange": {
"highFrequency": 3630000000,
"lowFrequency": 3620000000
}
}
}
]
}
<7>19:36:05.922 Sas.cpp post 12951 [36;1mDBG[0m [
"No data received",
"No data received",
"No data received"
]
<7>19:36:05.922 Sas.cpp post 12951 [36;1mDBG[0m null
<3>19:36:05.922 Grant.cpp grant_main_ 12951 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<3>19:36:05.922 Grant.cpp grant_main_ 12951 [31;1mERR[0m Grant error detail: No data received
<3>19:36:05.922 Grant.cpp grant_main_ 12951 [31;1mERR[0m Grant error detail: No data received
<3>19:36:05.922 Grant.cpp grant_main_ 12951 [31;1mERR[0m Grant error detail: No data received
<7>19:36:05.922 Grant.cpp grant_main_ 12951 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:36:05.922 CbrsDaemon.cpp onLoop 12951 [34;1mINF[0m Listening for 2 seconds
<6>19:36:08.949 CbrsDaemon.cpp parseTree 12951 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:36:08.952 Sas.cpp post 12951 [36;1mDBG[0m {
"grantRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"operationParam": {
"maxEirp": 0,
"operationFrequencyRange": {
"highFrequency": 3630000000,
"lowFrequency": 3620000000
}
}
}
]
}
<7>19:36:15.122 Sas.cpp post 12951 [36;1mDBG[0m [
"No data received",
"No data received",
"No data received"
]
<7>19:36:15.122 Sas.cpp post 12951 [36;1mDBG[0m null
<3>19:36:15.122 Grant.cpp grant_main_ 12951 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<3>19:36:15.122 Grant.cpp grant_main_ 12951 [31;1mERR[0m Grant error detail: No data received
<3>19:36:15.122 Grant.cpp grant_main_ 12951 [31;1mERR[0m Grant error detail: No data received
<3>19:36:15.122 Grant.cpp grant_main_ 12951 [31;1mERR[0m Grant error detail: No data received
<7>19:36:15.122 Grant.cpp grant_main_ 12951 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:36:15.123 CbrsDaemon.cpp onLoop 12951 [34;1mINF[0m Listening for 2 seconds
<6>19:36:18.147 CbrsDaemon.cpp parseTree 12951 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:36:18.150 Sas.cpp post 12951 [36;1mDBG[0m {
"grantRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"operationParam": {
"maxEirp": 0,
"operationFrequencyRange": {
"highFrequency": 3630000000,
"lowFrequency": 3620000000
}
}
}
]
}
<6>19:36:24.055 Daemon.cpp signalTreatment 12951 [34;1mINF[0m Received signal: 2.
<7>19:36:24.110 Sas.cpp post 12951 [36;1mDBG[0m [
"No data received",
"No data received",
"No data received"
]

9.20 Log file for test case ID: WINNF.FT.C.HBT.10


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>19:38:12.818 Sas.cpp post 13167 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>19:38:12.818 CbrsDaemon.cpp onLoop 13167 [34;1mINF[0m Listening for 2 seconds
<7>19:38:12.819 SpvLaunchdProxy.cpp create 13167 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>19:38:12.820 SpvLaunchdProxy.cpp create 13167 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>19:38:12.820 SpvLaunchdProxy.cpp initSpvLaunchdProxy 13167 [36;1mDBG[0m SpvLaunchd
is running.
<7>19:38:12.820 SpvLaunchdProxy.cpp logDBusMessage 13167 [36;1mDBG[0m
handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.89 serial=2
path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired;
signature=s
<7>19:38:12.820 SpvLaunchdProxy.cpp dbusHandler 13167 [36;1mDBG[0m NameAcquired:
:1.89
<7>19:38:12.820 SpvLaunchdProxy.cpp dbusHandler 13167 [36;1mDBG[0m Connection
name: :1.89
<6>19:38:13.842 CbrsDaemon.cpp parseTree 13167 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>19:38:13.845 CbrsDaemon.cpp onLoop 13167 [34;1mINF[0m Listening for 2 seconds
<6>19:38:16.872 CbrsDaemon.cpp parseTree 13167 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:38:16.874 CbrsDaemon.cpp persistEntities 13167 [36;1mDBG[0m Grant for cell 0,
belonging to eNB 1 created.
<7>19:38:16.875 Sas.cpp post 13167 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>19:38:16.879 Sas.cpp post 13167 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T17:38:16Z",
   "grantId": "350497071",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>19:38:16.879 Sas.cpp post 13167 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "350497071",
   "operationState": "GRANTED"
  }
 ]
}
<7>19:38:16.922 Sas.cpp post 13167 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "350497071",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:41:36Z"
  }
 ]
}
<6>19:38:16.923 CbrsDaemon.cpp onLoop 13167 [34;1mINF[0m Listening for 2 seconds
<7>19:38:16.923 SpvLaunchdProxy.cpp logDBusMessage 13167 [36;1mDBG[0m handleRequest:
signal sender=:1.0 -> dest=(null) serial=131 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StartProcess; signature=s
<7>19:38:16.923 SpvLaunchdProxy.cpp logActiveEnbs 13167 [36;1mDBG[0m Dump activeEnbs_
map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"in
valid_cfg":"","state":"CONNECTED"}]}
<6>19:38:17.948 CbrsDaemon.cpp parseTree 13167 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:38:17.950 Sas.cpp post 13167 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "350497071",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:38:17.953 Sas.cpp post 13167 [36;1mDBG[0m {
```

```
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "350497071",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:41:37Z"
  }
 ]
}
<6>19:38:17.953 ManagerEnb.cpp command 13167 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>19:38:17.954 CbrsDaemon.cpp onLoop 13167 [34;1mINF[0m Listening for 2 seconds
<6>19:38:18.093 Enb.cpp onData 13198 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>19:38:20.981 CbrsDaemon.cpp parseTree 13167 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:38:20.984 Sas.cpp post 13167 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "350497071",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:38:27.156 Sas.cpp post 13167 [36;1mDBG[0m [
 "No data received",
 "No data received",
 "No data received"
]
<7>19:38:27.156 Sas.cpp post 13167 [36;1mDBG[0m null
<3>19:38:27.156 Grant.cpp grant_main_ 13167 [31;1mERR[0m Heartbeat procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, grantId 350497071
<3>19:38:27.156 Grant.cpp grant_main_ 13167 [31;1mERR[0m Heartbeat error detail: No data received
<3>19:38:27.156 Grant.cpp grant_main_ 13167 [31;1mERR[0m Heartbeat error detail: No data received
<3>19:38:27.156 Grant.cpp grant_main_ 13167 [31;1mERR[0m Heartbeat error detail: No data received
<7>19:38:27.156 Grant.cpp grant_main_ 13167 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:38:27.156 ManagerEnb.cpp command 13167 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 0
<6>19:38:27.156 CbrsDaemon.cpp onLoop 13167 [34;1mINF[0m Listening for 2 seconds
<6>19:38:27.256 Enb.cpp onData 13198 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>19:38:30.183 CbrsDaemon.cpp parseTree 13167 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:38:30.186 Sas.cpp post 13167 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>19:38:36.359 Sas.cpp post 13167 [36;1mDBG[0m [
 "No data received",
 "No data received",
 "No data received"
]
<7>19:38:36.359 Sas.cpp post 13167 [36;1mDBG[0m null
<3>19:38:36.359 Grant.cpp grant_main_ 13167 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<3>19:38:36.359 Grant.cpp grant_main_ 13167 [31;1mERR[0m Grant error detail: No data received
<3>19:38:36.359 Grant.cpp grant_main_ 13167 [31;1mERR[0m Grant error detail: No data received
<3>19:38:36.359 Grant.cpp grant_main_ 13167 [31;1mERR[0m Grant error detail: No data received
<7>19:38:36.359 Grant.cpp grant_main_ 13167 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:38:36.359 CbrsDaemon.cpp onLoop 13167 [34;1mINF[0m Listening for 2 seconds
<6>19:38:39.383 CbrsDaemon.cpp parseTree 13167 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:38:39.386 Sas.cpp post 13167 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<6>19:38:45.220 Daemon.cpp signalTreatment 13167 [34;1mINF[0m Received signal: 2.
<7>19:38:45.275 Sas.cpp post 13167 [36;1mDBG[0m [
 "No data received",
 "No data received",
 "No data received"
]
```

9.21 Log file for test case ID: WINNF.FT.C.HBT.11


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT",
    "RECEIVED_POWER_WITHOUT_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>20:14:35.260 Sas.cpp post 15851 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>20:14:35.261 CbrsDaemon.cpp onLoop 15851 [34;1mINF[0m Listening for 59 seconds
<7>20:14:35.262 SpvLaunchdProxy.cpp create 15851 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>20:14:35.262 SpvLaunchdProxy.cpp create 15851 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>20:14:35.262 SpvLaunchdProxy.cpp initSpvLaunchdProxy 15851 [36;1mDBG[0m SpvLaunchd is running.
<7>20:14:35.262 SpvLaunchdProxy.cpp logDBusMessage 15851 [36;1mDBG[0m handleRequest: signal
sender=org.freedesktop.DBus -> dest=:1.117 serial=2 path=/org/freedesktop/DBus;
interface=org.freedesktop.DBus; member=NameAcquired; signature=s
<7>20:14:35.262 SpvLaunchdProxy.cpp dbusHandler 15851 [36;1mDBG[0m NameAcquired: :1.117
<7>20:14:35.262 SpvLaunchdProxy.cpp dbusHandler 15851 [36;1mDBG[0m Connection name: :1.117
<6>20:14:36.285 CbrsDaemon.cpp parseTree 15851 [34;1mINF[0m Found CBRS Cell: cell_id 0,
earfcn_dl 55990
<6>20:14:36.287 CbrsDaemon.cpp onLoop 15851 [34;1mINF[0m Listening for 59 seconds
<7>20:14:40.550 SpvLaunchdProxy.cpp logDBusMessage 15851 [36;1mDBG[0m handleRequest: signal
sender=:1.0 -> dest=(null) serial=207 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StartProcess; signature=s
<7>20:14:40.550 SpvLaunchdProxy.cpp logActiveEnbs 15851 [36;1mDBG[0m Dump activeEnbs_ map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"invalid_
cfg":"","state":"CONNECTED"}]}
<6>20:14:41.576 CbrsDaemon.cpp parseTree 15851 [34;1mINF[0m Found CBRS Cell: cell_id 0,
earfcn_dl 55990
<7>20:14:41.579 CbrsDaemon.cpp persistEntities 15851 [36;1mDBG[0m Grant for cell
0, belonging to eNB 1 created.
<6>20:14:41.580 ManagerCbsd.cpp command 15851 [34;1mINF[0m Send
command to CBSD on fe80::72b3:d5ff:fe29:c2f1:
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":15589808
81,"user":"user"}, with timeout of 60
<6>20:14:41.580 ManagerCbsd.cpp getResponseFromReque 15851 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Send (timeout 60 seconds):
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":15589808
81,"user":"user"}
<6>20:14:41.609 ManagerCbsd.cpp getResponseFromReque 15851 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Socket response received (168470 bytes)
<7>20:14:41.612 Sas.cpp post 15851 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -98
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -95
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3630000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3640000000,
      "measRcvdPower": -100
     },
     {
```

```
        "measBandwidth": 10000000,
        "measFrequency": 3650000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3660000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3670000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3680000000,
        "measRcvdPower": -96
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3690000000,
        "measRcvdPower": -99
       }
      ]
     },
     "operationParam": {
      "maxEirp": 0,
      "operationFrequencyRange": {
       "highFrequency": 3630000000,
       "lowFrequency": 3620000000
      }
     }
    }
   ]
}
<7>20:14:41.618 Sas.cpp          post            15851 [36;1mDBG[0m {
  "grantResponse": [
    {
     "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
     "channelType": "GAA",
     "grantExpireTime": "2019-05-27T18:20:41Z",
     "grantId": "224492604",
     "heartbeatInterval": 60,
     "response": {
      "responseCode": 0
     }
    }
   ]
}
<7>20:14:41.618 Sas.cpp          post            15851 [36;1mDBG[0m {
  "heartbeatRequest": [
    {
     "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
     "grantId": "224492604",
     "operationState": "GRANTED"
    }
   ]
}
<7>20:14:41.661 Sas.cpp          post            15851 [36;1mDBG[0m {
  "heartbeatResponse": [
    {
     "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
     "grantId": "224492604",
     "response": {
      "responseCode": 0
     },
     "transmitExpireTime": "2019-05-27T18:18:01Z"
    }
   ]
}
<6>20:14:41.661 ManagerEnb.cpp   command        15851 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>20:14:41.662 CbrsDaemon.cpp   onLoop         15851 [34;1mINF[0m Listening for 59 seconds
<6>20:14:42.109 Enb.cpp          onData         15882 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>20:15:41.746 CbrsDaemon.cpp   parseTree      15851 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>20:15:41.749 Sas.cpp          post            15851 [36;1mDBG[0m {
  "heartbeatRequest": [
    {
     "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
     "grantId": "224492604",
     "operationState": "AUTHORIZED"
    }
   ]
}
<7>20:15:41.753 Sas.cpp          post            15851 [36;1mDBG[0m {
  "heartbeatResponse": [
    {
     "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
     "grantId": "224492604",
     "response": {
      "responseCode": 0
     },
     "transmitExpireTime": "2019-05-27T18:19:01Z"
    }
   ]
}
<6>20:15:41.753 ManagerEnb.cpp   command        15851 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>20:15:41.753 CbrsDaemon.cpp   onLoop         15851 [34;1mINF[0m Listening for 59 seconds
<6>20:15:41.853 Enb.cpp          onData         15882 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>20:16:41.838 CbrsDaemon.cpp   parseTree      15851 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>20:16:41.840 Sas.cpp          post            15851 [36;1mDBG[0m {
  "heartbeatRequest": [
    {
     "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
     "grantId": "224492604",
     "operationState": "AUTHORIZED"
    }
   ]
}
<7>20:16:41.844 Sas.cpp          post            15851 [36;1mDBG[0m {
  "heartbeatResponse": [
    {
     "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
     "grantId": "224492604",
     "response": {
      "responseCode": 0
     },
     "transmitExpireTime": "2019-05-27T18:20:01Z"
    }
   ]
}
<6>20:16:41.844 ManagerEnb.cpp   command        15851 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>20:16:41.845 CbrsDaemon.cpp   onLoop         15851 [34;1mINF[0m Listening for 59 seconds
<6>20:16:41.944 Enb.cpp          onData         15882 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>20:17:41.926 CbrsDaemon.cpp   parseTree      15851 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>20:17:41.929 Sas.cpp          post            15851 [36;1mDBG[0m {
```

```
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "224492604",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>20:17:41.932 Sas.cpp          post            15851 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "224492604",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:20:41Z"
  }
 ]
}
<6>20:17:41.932 ManagerEnb.cpp     command          15851 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>20:17:41.933 CbrsDaemon.cpp     onLoop           15851 [34;1mINF[0m Listening for 59 seconds
<6>20:17:42.033 Enb.cpp          onData           15882 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>20:18:42.017 CbrsDaemon.cpp     parseTree        15851 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>20:18:42.020 Sas.cpp          post            15851 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "224492604",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>20:18:42.023 Sas.cpp          post            15851 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "224492604",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:20:41Z"
  }
 ]
}
<6>20:18:42.023 ManagerEnb.cpp     command          15851 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>20:18:42.024 CbrsDaemon.cpp     onLoop           15851 [34;1mINF[0m Listening for 59 seconds
<6>20:18:42.124 Enb.cpp          onData           15882 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>20:19:42.105 CbrsDaemon.cpp     parseTree        15851 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>20:19:42.108 Sas.cpp          post            15851 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "224492604",
   "grantRenew": true,
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>20:19:42.113 Sas.cpp          post            15851 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantExpireTime": "2019-05-27T18:25:42Z",
   "grantId": "224492604",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:23:02Z"
  }
 ]
}
```

9.22 Log file for test case ID: WINNF.FT.C.MES.1


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT",
    "RECEIVED_POWER_WITHOUT_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>19:45:36.502 Sas.cpp post 13742 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReportConfig": [
    "RECEIVED_POWER_WITHOUT_GRANT"
   ],
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>19:45:36.502 CbrsDaemon.cpp onLoop 13742 [34;1mINF[0m Listening for 2 seconds
<7>19:45:36.503 SpvLaunchdProxy.cpp create 13742 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>19:45:36.503 SpvLaunchdProxy.cpp create 13742 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>19:45:36.503 SpvLaunchdProxy.cpp initSpvLaunchdProxy 13742 [36;1mDBG[0m SpvLaunchd
is running.
<7>19:45:36.503 SpvLaunchdProxy.cpp logDBusMessage 13742 [36;1mDBG[0m
handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.94 serial=2
path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired;
signature=s
<7>19:45:36.503 SpvLaunchdProxy.cpp dbusHandler 13742 [36;1mDBG[0m NameAcquired:
:1.94
<7>19:45:36.503 SpvLaunchdProxy.cpp dbusHandler 13742 [36;1mDBG[0m Connection
name: :1.94
<6>19:45:37.526 CbrsDaemon.cpp parseTree 13742 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>19:45:37.528 CbrsDaemon.cpp onLoop 13742 [34;1mINF[0m Listening for 2 seconds
<7>19:45:39.469 SpvLaunchdProxy.cpp logDBusMessage 13742 [36;1mDBG[0m handleRequest:
signal sender=:1.0 -> dest=(null) serial=144 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StartProcess; signature=s
<7>19:45:39.469 SpvLaunchdProxy.cpp logActiveEnbs 13742 [36;1mDBG[0m Dump activeEnbs_
map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"in
valid_cfg":"","state":"CONNECTED"}]}
<6>19:45:40.491 CbrsDaemon.cpp parseTree 13742 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:45:40.494 CbrsDaemon.cpp persistEntities 13742 [36;1mDBG[0m Grant for cell 0,
belonging to eNB 1 created.
<6>19:45:40.495 ManagerCbsd.cpp command 13742 [34;1mINF[0m Send command to CBSD
on fe80::72b3:d5ff:fe29:c2f1:
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558979140,"user":
"user"}, with timeout of 30
<6>19:45:40.495 ManagerCbsd.cpp getResponseFromReque 13742 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Send (timeout 30 seconds):
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558979140,"user":
"user"}
<6>19:45:40.650 ManagerCbsd.cpp getResponseFromReque 13742 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Socket response received (166129 bytes)
<7>19:45:40.653 Sas.cpp post 13742 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3630000000,
      "measRcvdPower": -100
     },
```

```
        {
          "measBandwidth": 10000000,
          "measFrequency": 3640000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3650000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3660000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3670000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3680000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3690000000,
          "measRcvdPower": -100
        }
      ]
    },
    "operationParam": {
      "maxEirp": 0,
      "operationFrequencyRange": {
        "highFrequency": 3630000000,
        "lowFrequency": 3620000000
      }
    }
  }
 ]
}
<7>19:45:40.659 Sas.cpp          post          13742 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T17:45:40Z",
   "grantId": "672187239",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>19:45:40.659 Sas.cpp          post          13742 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "672187239",
   "operationState": "GRANTED"
  }
 ]
}
<7>19:45:40.699 Sas.cpp          post          13742 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "672187239",
   "response": {
    "responseCode": 501
   },
   "transmitExpireTime": "2019-05-27T17:45:40Z"
  }
 ]
}
```

```
<3>19:45:40.699 Grant.cpp        grant_main_       13742 [31;1mERR[0m Heartbeat procedure failed
for CBSD XM2-X19AX35M2Mock-SAS1012482003, grantId 672187239
<7>19:45:40.699 Grant.cpp        grant_main_       13742 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:45:40.700 CbrsDaemon.cpp   onLoop            13742 [34;1mINF[0m Listening for 2 seconds
<6>19:45:43.724 CbrsDaemon.cpp   parseTree         13742 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:45:43.726 Sas.cpp          post          13742 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3630000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3640000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3650000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3660000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3670000000,
      "measRcvdPower": -100
     },
```

```
        {
          "measBandwidth": 10000000,
          "measFrequency": 3680000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3690000000,
          "measRcvdPower": -100
        }
       ]
      },
      "operationParam": {
       "maxEirp": 0,
       "operationFrequencyRange": {
        "highFrequency": 3630000000,
        "lowFrequency": 3620000000
       }
      }
     }
    ]
}
<7>19:45:43.727 Sas.cpp          post            13742 [36;1mDBG[0m {
  "grantResponse": [
    {
      "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
      "response": {
        "responseCode": 400
      }
    }
  ]
}
<3>19:45:43.727 Grant.cpp        grant_main_     13742 [31;1mERR[0m Grant procedure failed
for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>19:45:43.727 Grant.cpp        grant_main_     13742 [36;1mDBG[0m ERROR state reset to
IDLE
<6>19:45:43.728 CbrsDaemon.cpp   onLoop          13742 [34;1mINF[0m Listening for 2 seconds
<6>19:45:46.755 CbrsDaemon.cpp   parseTree       13742 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>19:45:46.758 Sas.cpp          post            13742 [36;1mDBG[0m {
  "grantRequest": [
    {
      "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
      "measReport": {
        "rcvdPowerMeasReports": [
          {
            "measBandwidth": 10000000,
            "measFrequency": 3550000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3560000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3570000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3580000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3590000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3600000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3610000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3620000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3630000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3640000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3650000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3660000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3670000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3680000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3690000000,
            "measRcvdPower": -100
          }
        ]
      },
      "operationParam": {
        "maxEirp": 0,
        "operationFrequencyRange": {
          "highFrequency": 3630000000,
          "lowFrequency": 3620000000
        }
      }
    }
  ]
}
<7>19:45:46.759 Sas.cpp          post            13742 [36;1mDBG[0m {
  "grantResponse": [
    {
      "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
      "response": {
        "responseCode": 400
      }
    }
  ]
}
<3>19:45:46.759 Grant.cpp        grant_main_     13742 [31;1mERR[0m Grant procedure failed for
CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>19:45:46.759 Grant.cpp        grant_main_     13742 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:45:46.760 CbrsDaemon.cpp   onLoop          13742 [34;1mINF[0m Listening for 2 seconds
<6>19:45:49.787 CbrsDaemon.cpp   parseTree       13742 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:45:49.789 Sas.cpp          post            13742 [36;1mDBG[0m {
  "grantRequest": [
    {
      "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
      "measReport": {
        "rcvdPowerMeasReports": [
          {
            "measBandwidth": 10000000,
            "measFrequency": 3550000000,
            "measRcvdPower": -100
```

```
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -100
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3630000000,
      "measRcvdPower": -100
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3640000000,
      "measRcvdPower": -100
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3650000000,
      "measRcvdPower": -100
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3660000000,
      "measRcvdPower": -100
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3670000000,
      "measRcvdPower": -100
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3680000000,
      "measRcvdPower": -100
    },
    {
      "measBandwidth": 10000000,
      "measFrequency": 3690000000,
      "measRcvdPower": -100
    }
   ]
  },
  "operationParam": {
   "maxEirp": 0,
   "operationFrequencyRange": {
    "highFrequency": 3630000000,
    "lowFrequency": 3620000000
   }
  }
 }
}
<7>19:45:49.790 Sas.cpp          post             13742 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 400
   }
  }
 ]
}
<3>19:45:49.790 Grant.cpp        grant_main_      13742 [31;1mERR[0m Grant procedure failed for
CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>19:45:49.790 Grant.cpp        grant_main_      13742 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:45:49.791 CbrsDaemon.cpp   onLoop           13742 [34;1mINF[0m Listening for 2 seconds
<6>19:45:52.818 CbrsDaemon.cpp   parseTree        13742 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:45:52.820 Sas.cpp          post             13742 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3630000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3640000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3650000000,
      "measRcvdPower": -100
     },
```

```
        {
          "measBandwidth": 10000000,
          "measFrequency": 3660000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3670000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3680000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3690000000,
          "measRcvdPower": -100
        }
       ]
      },
      "operationParam": {
       "maxEirp": 0,
       "operationFrequencyRange": {
        "highFrequency": 3630000000,
        "lowFrequency": 3620000000
       }
      }
     }
    ]
}
<7>19:45:52.821 Sas.cpp          post             13742 [36;1mDBG[0m {
  "grantResponse": [
    {
      "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
      "response": {
       "responseCode": 400
      }
     }
    ]
}
<3>19:45:52.821 Grant.cpp        grant_main_      13742 [31;1mERR[0m Grant procedure failed
for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>19:45:52.821 Grant.cpp        grant_main_      13742 [36;1mDBG[0m ERROR state reset to
IDLE
<6>19:45:52.822 CbrsDaemon.cpp   onLoop           13742 [34;1mINF[0m Listening for 2 seconds
<6>19:45:55.849 CbrsDaemon.cpp   parseTree        13742 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>19:45:55.851 Sas.cpp          post             13742 [36;1mDBG[0m {
  "grantRequest": [
    {
      "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
      "measReport": {
       "rcvdPowerMeasReports": [
        {
          "measBandwidth": 10000000,
          "measFrequency": 3550000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3560000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3570000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3580000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3590000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3600000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3610000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3620000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3630000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3640000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3650000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3660000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3670000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3680000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3690000000,
          "measRcvdPower": -100
        }
       ]
      },
      "operationParam": {
       "maxEirp": 0,
       "operationFrequencyRange": {
        "highFrequency": 3630000000,
        "lowFrequency": 3620000000
       }
      }
     }
    ]
}
<7>19:45:55.852 Sas.cpp          post             13742 [36;1mDBG[0m {
  "grantResponse": [
    {
      "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
      "response": {
       "responseCode": 400
      }
     }
    ]
}
<3>19:45:55.852 Grant.cpp        grant_main_      13742 [31;1mERR[0m Grant procedure failed for
CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>19:45:55.852 Grant.cpp        grant_main_      13742 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:45:55.853 CbrsDae
<6>19:45:58.877 CbrsDaemon.cpp   parseTree        13742 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:45:58.880 Sas.cpp          post             13742 [36;1mDBG[0m {mon.cpp      onLoop
13742 [34;1mINF[0m Listening for 2 seconds
```

```
"grantRequest": [
 {
  "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
  "measReport": {
   "rcvdPowerMeasReports": [
    {
     "measBandwidth": 10000000,
     "measFrequency": 3550000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3560000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3570000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3580000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3590000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3600000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3610000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3620000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3630000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3640000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3650000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3660000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3670000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3680000000,
     "measRcvdPower": -100
    },
    {
     "measBandwidth": 10000000,
     "measFrequency": 3690000000,
     "measRcvdPower": -100
    }
   ]
  },
  "operationParam": {
   "maxEirp": 0,
   "operationFrequencyRange": {
    "highFrequency": 3630000000,
    "lowFrequency": 3620000000
   }
  }
 }
]
}
<7>19:45:58.881 Sas.cpp          post            13742 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 400
   }
  }
 ]
}
<3>19:45:58.881 Grant.cpp        grant_main_     13742 [31;1mERR[0m Grant procedure failed for
CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>19:45:58.881 Grant.cpp        grant_main_     13742 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:45:58.881 CbrsDaemon.cpp   onLoop          13742 [34;1mINF[0m Listening for 2 seconds
<6>19:46:01.905 CbrsDaemon.cpp   parseTree       13742 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:46:01.908 Sas.cpp          post            13742 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3630000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3640000000,
      "measRcvdPower": -100
```

```
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3650000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3660000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3670000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3680000000,
          "measRcvdPower": -100
        },
        {
          "measBandwidth": 10000000,
          "measFrequency": 3690000000,
          "measRcvdPower": -100
        }
      ]
     },
     "operationParam": {
      "maxEirp": 0,
      "operationFrequencyRange": {
       "highFrequency": 3630000000,
       "lowFrequency": 3620000000
      }
     }
    }
   ]
}
<7>19:46:01.909 Sas.cpp          post             13742 [36;1mDBG[0m {
  "grantResponse": [
    {
      "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
     "response": {
       "responseCode": 400
     }
    }
   ]
}
<3>19:46:01.909 Grant.cpp        grant_main_      13742 [31;1mERR[0m Grant procedure failed
for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>19:46:01.909 Grant.cpp        grant_main_      13742 [36;1mDBG[0m ERROR state reset to
IDLE
<6>19:46:01.910 CbrsDaemon.cpp   onLoop           13742 [34;1mINF[0m Listening for 2 seconds
<6>19:46:04.936 CbrsDaemon.cpp   parseTree        13742 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>19:46:04.938 Sas.cpp          post             13742 [36;1mDBG[0m {
  "grantRequest": [
    {
      "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
      "measReport": {
        "rcvdPowerMeasReports": [
          {
            "measBandwidth": 10000000,
            "measFrequency": 3550000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3560000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3570000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3580000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3590000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3600000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3610000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3620000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3630000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3640000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3650000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3660000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3670000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3680000000,
            "measRcvdPower": -100
          },
          {
            "measBandwidth": 10000000,
            "measFrequency": 3690000000,
            "measRcvdPower": -100
          }
        ]
      },
      "operationParam": {
        "maxEirp": 0,
        "operationFrequencyRange": {
          "highFrequency": 3630000000,
          "lowFrequency": 3620000000
        }
      }
    }
  ]
}
<7>19:46:04.940 Sas.cpp          post             13742 [36;1mDBG[0m {
  "grantResponse": [
    {
      "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
      "response": {
        "responseCode": 400
      }
    }
  ]
}
<3>19:46:04.940 Grant.cpp        grant_main_      13742 [31;1mERR[0m Grant procedure failed for
CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>19:46:04.940 Grant.cpp        grant_main_      13742 [36;1mDBG[0m ERROR state reset to IDLE
```

```
<6>19:46:04.940 CbrsDaemon.cpp      onLoop            13742 [34;1mINF[0m Listening for 2 seconds
<6>19:46:07.964 CbrsDaemon.cpp      parseTree         13742 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>19:46:07.966 Sas.cpp             post              13742 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3630000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3640000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3650000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3660000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3670000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3680000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3690000000,
      "measRcvdPower": -100
     }
    ]
   },
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>19:46:07.967 Sas.cpp             post              13742 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 400
   }
  }
 ]
}
<3>19:46:07.967 Grant.cpp           grant_main_       13742 [31;1mERR[0m Grant procedure failed for
CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>19:46:07.967 Grant.cpp           grant_main_       13742 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:46:07.968 CbrsDaemon.cpp      onLoop            13742 [34;1mINF[0m Listening for 2 seconds
<6>19:46:10.992 CbrsDaemon.cpp      parseTree         13742 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:46:10.994 Sas.cpp             post              13742 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
```

```
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3630000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3640000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3650000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3660000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3670000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3680000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3690000000,
        "measRcvdPower": -100
      }
     ]
    },
    "operationParam": {
     "maxEirp": 0,
     "operationFrequencyRange": {
      "highFrequency": 3630000000,
      "lowFrequency": 3620000000
     }
    }
   }
  ]
}
<7>19:46:10.996 Sas.cpp          post            13742 [36;1mDBG[0m {
  "grantResponse": [
   {
    "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
    "response": {
     "responseCode": 400
    }
   }
  ]
}
<3>19:46:10.996 Grant.cpp        grant_main_     13742 [31;1mERR[0m Grant procedure failed
for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>19:46:10.996 Grant.cpp        grant_main_     13742 [36;1mDBG[0m ERROR state reset to
IDLE
<6>19:46:10.996 CbrsDaemon.cpp   onLoop          13742 [34;1mINF[0m Listening for 2 seconds
<6>19:46:14.023 CbrsDaemon.cpp   parseTree       13742 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>19:46:14.025 Sas.cpp          post            13742 [36;1mDBG[0m {
  "grantRequest": [
   {
    "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
    "measReport": {
     "rcvdPowerMeasReports": [
      {
       "measBandwidth": 10000000,
       "measFrequency": 3550000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3560000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3570000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3580000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3590000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3600000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3610000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3620000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3630000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3640000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3650000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3660000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3670000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3680000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3690000000,
       "measRcvdPower": -100
      }
     ]
    },
    "operationParam": {
     "maxEirp": 0,
     "operationFrequencyRange": {
      "highFrequency": 3630000000,
      "lowFrequency": 3620000000
     }
    }
   }
  ]
}
```

<7>19:46:14.026 Sas.cpp post 13742 [36;1mDBG[0m {
"grantResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"response": {
"responseCode": 400
}
}
]
}
<3>19:46:14.027 Grant.cpp grant_main_ 13742 [31;1mERR[0m Grant procedure failed for CBSD XM2-X19AX35M2Mock-SAS1012482003, cell 0 eNB 1
<7>19:46:14.027 Grant.cpp grant_main_ 13742 [36;1mDBG[0m ERROR state reset to IDLE
<6>19:46:14.027 CbrsDaemon.cpp onLoop 13742 [34;1mINF[0m Listening for 2 seconds
<6>19:46:17.054 CbrsDaemon.cpp parseTree 13742 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:46:17.056 Sas.cpp post 13742 [36;1mDBG[0m {
"grantRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"measReport": {
"rcvdPowerMeasReports": [
{
"measBandwidth": 10000000,
"measFrequency": 3550000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3560000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3570000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3580000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3590000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3600000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3610000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3620000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3630000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3640000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3650000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3660000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3670000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3680000000,
"measRcvdPower": -100
},
{
"measBandwidth": 10000000,
"measFrequency": 3690000000,
"measRcvdPower": -100
}
]
},
"operationParam": {
"maxEirp": 0,
"operationFrequencyRange": {
"highFrequency": 3630000000,
"lowFrequency": 3620000000
}
}
}
]
}
<7>19:46:17.057 Sas.cpp post 13742 [36;1mDBG[0m {
"grantResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"response": {
"responseCode": 400
}
}
]
}

9.23 Log file for test case ID: WINNF.FT.C.MES.3


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT",
    "RECEIVED_POWER_WITHOUT_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>19:48:40.970 Sas.cpp post 13943 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>19:48:40.971 CbrsDaemon.cpp onLoop 13943 [34;1mINF[0m Listening for 2 seconds
<7>19:48:40.971 SpvLaunchdProxy.cpp create 13943 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>19:48:40.971 SpvLaunchdProxy.cpp create 13943 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>19:48:40.972 SpvLaunchdProxy.cpp initSpvLaunchdProxy 13943 [36;1mDBG[0m SpvLaunchd
is running.
<7>19:48:40.972 SpvLaunchdProxy.cpp logDBusMessage 13943 [36;1mDBG[0m
handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.96 serial=2
path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired;
signature=s
<7>19:48:40.972 SpvLaunchdProxy.cpp dbusHandler 13943 [36;1mDBG[0m NameAcquired:
:1.96
<7>19:48:40.972 SpvLaunchdProxy.cpp dbusHandler 13943 [36;1mDBG[0m Connection
name: :1.96
<6>19:48:41.994 CbrsDaemon.cpp parseTree 13943 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>19:48:41.997 CbrsDaemon.cpp onLoop 13943 [34;1mINF[0m Listening for 2 seconds
<7>19:48:43.910 SpvLaunchdProxy.cpp logDBusMessage 13943 [36;1mDBG[0m handleRequest:
signal sender=:1.0 -> dest=(null) serial=150 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StartProcess; signature=s
<7>19:48:43.910 SpvLaunchdProxy.cpp logActiveEnbs 13943 [36;1mDBG[0m Dump activeEnbs_
map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"in
valid_cfg":"","state":"CONNECTED"}]}
<6>19:48:44.932 CbrsDaemon.cpp parseTree 13943 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:48:44.935 CbrsDaemon.cpp persistEntities 13943 [36;1mDBG[0m Grant for cell 0,
belonging to eNB 1 created.
<6>19:48:44.936 ManagerCbsd.cpp command 13943 [34;1mINF[0m Send command to CBSD
on fe80::72b3:d5ff:fe29:c2f1:
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558979324,"user":
"user"}, with timeout of 30
<6>19:48:44.936 ManagerCbsd.cpp getResponseFromReque 13943 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Send (timeout 30 seconds):
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558979324,"user":
"user"}
<6>19:48:44.968 ManagerCbsd.cpp getResponseFromReque 13943 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Socket response received (198746 bytes)
<7>19:48:44.971 Sas.cpp post 13943 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -96
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -99
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -70
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
     },
     {
```

```
        "measBandwidth": 10000000,
        "measFrequency": 3630000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3640000000,
        "measRcvdPower": -99
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3650000000,
        "measRcvdPower": -68
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3660000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3670000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3680000000,
        "measRcvdPower": -94
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3690000000,
        "measRcvdPower": -97
       }
      ]
     },
     "operationParam": {
      "maxEirp": 0,
      "operationFrequencyRange": {
       "highFrequency": 3630000000,
       "lowFrequency": 3620000000
      }
     }
    }
   ]
}
<7>19:48:44.978 Sas.cpp          post            13943 [36;1mDBG[0m {
  "grantResponse": [
    {
     "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
     "channelType": "GAA",
     "grantExpireTime": "2019-06-03T17:48:44Z",
     "grantId": "308900511",
     "heartbeatInterval": 60,
     "measReportConfig": [
      "RECEIVED_POWER_WITH_GRANT"
     ],
     "response": {
      "responseCode": 0
     }
    }
   ]
}
<7>19:48:44.978 Sas.cpp          post            13943 [36;1mDBG[0m {
  "heartbeatRequest": [
    {
     "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
     "grantId": "308900511",
     "measReport": {
      "rcvdPowerMeasReports": [
       {
        "measBandwidth": 10000000,
        "measFrequency": 3550000000,
        "measRcvdPower": -96
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3560000000,
        "measRcvdPower": -99
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3570000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3580000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3590000000,
        "measRcvdPower": -70
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3600000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3610000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3620000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3630000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3640000000,
        "measRcvdPower": -99
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3650000000,
        "measRcvdPower": -68
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3660000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3670000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3680000000,
        "measRcvdPower": -94
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3690000000,
        "measRcvdPower": -97
       }
      ]
     },
     "operationState": "GRANTED"
    }
   ]
}
```

<7>19:48:45.022 Sas.cpp post 13943 [36;1mDBG[0m {
"heartbeatResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "308900511",
"response": {
"responseCode": 0
},
"transmitExpireTime": "2019-05-27T17:52:04Z"
}
]
}
<6>19:48:45.022 ManagerEnb.cpp command 13943 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>19:48:45.023 CbrsDaemon.cpp onLoop 13943 [34;1mINF[0m Listening for 2 seconds
<6>19:48:45.162 Enb.cpp onData 13974 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>19:48:48.050 CbrsDaemon.cpp parseTree 13943 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:48:48.052 Sas.cpp post 13943 [36;1mDBG[0m {
"heartbeatRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "308900511",
"operationState": "AUTHORIZED"
}
]
}
<7>19:48:48.056 Sas.cpp post 13943 [36;1mDBG[0m {
"heartbeatResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "308900511",
"response": {
"responseCode": 0
},
"transmitExpireTime": "2019-05-27T17:52:08Z"
}
]
}
<6>19:48:48.056 ManagerEnb.cpp command 13943 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>19:48:48.057 CbrsDaemon.cpp onLoop 13943 [34;1mINF[0m Listening for 2 seconds
<6>19:48:48.156 Enb.cpp onData 13974 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>19:48:51.080 CbrsDaemon.cpp parseTree 13943 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:48:51.083 Sas.cpp post 13943 [36;1mDBG[0m {
"heartbeatRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "308900511",
"operationState": "AUTHORIZED"
}
]
}
<7>19:48:51.087 Sas.cpp post 13943 [36;1mDBG[0m {
"heartbeatResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "308900511",
"response": {
"responseCode": 0
},
"transmitExpireTime": "2019-05-27T17:52:11Z"
}
]
}
<6>19:48:51.087 ManagerEnb.cpp command 13943 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>19:48:51.087 CbrsDaemon.cpp onLoop 13943 [34;1mINF[0m Listening for 2 seconds
<6>19:48:51.187 Enb.cpp onData 13974 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>19:48:54.111 CbrsDaemon.cpp parseTree 13943 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:48:54.114 Sas.cpp post 13943 [36;1mDBG[0m {
"heartbeatRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "308900511",
"operationState": "AUTHORIZED"
}
]
}
<7>19:48:54.118 Sas.cpp post 13943 [36;1mDBG[0m {
"heartbeatResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "308900511",
"response": {
"responseCode": 0
},
"transmitExpireTime": "2019-05-27T17:52:14Z"
}
]
}
<6>19:48:54.118 ManagerEnb.cpp command 13943 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>19:48:54.118 CbrsDaemon.cpp onLoop 13943 [34;1mINF[0m Listening for 2 seconds
<6>19:48:54.218 Enb.cpp onData 13974 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>19:48:57.142 CbrsDaemon.cpp parseTree 13943 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:48:57.145 Sas.cpp post 13943 [36;1mDBG[0m {
"heartbeatRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "308900511",
"operationState": "AUTHORIZED"
}
]
}
<7>19:48:57.149 Sas.cpp post 13943 [36;1mDBG[0m {
"heartbeatResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "308900511",
"response": {
"responseCode": 0
},
"transmitExpireTime": "2019-05-27T17:52:17Z"
}
]
}
<6>19:48:57.149 ManagerEnb.cpp command 13943 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>19:48:57.150 CbrsDaemon.cpp onLoop 13943 [34;1mINF[0m Listening for 2 seconds
<6>19:48:57.250 Enb.cpp onData 13974 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>19:49:00.173 CbrsDaemon.cpp parseTree 13943 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:49:00.176 Sas.cpp post 13943 [36;1mDBG[0m {
"heartbeatRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "308900511",
"operationState": "AUTHORIZED"
}
]
}
<7>19:49:00.177 Sas.cpp post 13943 [36;1mDBG[0m {
"heartbeatResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "308900511",
"response": {
"responseCode": 0
},
"transmitExpireTime": "2019-05-27T17:52:20Z"
}
]
}

```
<6>19:49:00.177 ManagerEnb.cpp      command            13943 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>19:49:00.177 CbrsDaemon.cpp      onLoop             13943 [34;1mINF[0m Listening for 2 seconds
<6>19:49:00.277 Enb.cpp             onData             13974 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<6>19:49:03.201 CbrsDaemon.cpp      parseTree          13943 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:49:03.204 Sas.cpp             post               13943 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "308900511",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:49:03.204 Sas.cpp             post               13943 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "308900511",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:52:23Z"
  }
 ]
}
```

9.24 Log file for test case ID: WINNF.FT.C.MES.4


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT",
    "RECEIVED_POWER_WITHOUT_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>19:51:44.093 Sas.cpp post 14145 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>19:51:44.093 CbrsDaemon.cpp onLoop 14145 [34;1mINF[0m Listening for 2 seconds
<7>19:51:44.094 SpvLaunchdProxy.cpp create 14145 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>19:51:44.094 SpvLaunchdProxy.cpp create 14145 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>19:51:44.094 SpvLaunchdProxy.cpp initSpvLaunchdProxy 14145 [36;1mDBG[0m SpvLaunchd
is running.
<7>19:51:44.094 SpvLaunchdProxy.cpp logDBusMessage 14145 [36;1mDBG[0m
handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.98 serial=2
path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired;
signature=s
<7>19:51:44.094 SpvLaunchdProxy.cpp dbusHandler 14145 [36;1mDBG[0m NameAcquired:
:1.98
<7>19:51:44.094 SpvLaunchdProxy.cpp dbusHandler 14145 [36;1mDBG[0m Connection
name: :1.98
<6>19:51:45.117 CbrsDaemon.cpp parseTree 14145 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>19:51:45.119 CbrsDaemon.cpp onLoop 14145 [34;1mINF[0m Listening for 2 seconds
<6>19:51:48.144 CbrsDaemon.cpp parseTree 14145 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:51:48.147 CbrsDaemon.cpp persistEntities 14145 [36;1mDBG[0m Grant for cell 0,
belonging to eNB 1 created.
<6>19:51:48.148 ManagerCbsd.cpp command 14145 [34;1mINF[0m Send command to CBSD
on fe80::72b3:d5ff:fe29:c2f1:
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558979508,"user":
"user"}, with timeout of 30
<6>19:51:48.148 ManagerCbsd.cpp getResponseFromReque 14145 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Send (timeout 30 seconds):
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558979508,"user":
"user"}
<6>19:51:48.180 ManagerCbsd.cpp getResponseFromReque 14145 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Socket response received (198589 bytes)
<7>19:51:48.184 Sas.cpp post 14145 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -96
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -95
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3630000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3640000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3650000000,
      "measRcvdPower": -99
```

```
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3660000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3670000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3680000000,
        "measRcvdPower": -95
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3690000000,
        "measRcvdPower": -97
      }
     ]
    },
    "operationParam": {
     "maxEirp": 0,
     "operationFrequencyRange": {
      "highFrequency": 3630000000,
      "lowFrequency": 3620000000
     }
    }
   }
  ]
}
<7>19:51:48.190 Sas.cpp          post            14145 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T17:51:48Z",
   "grantId": "502757752",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>19:51:48.190 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "operationState": "GRANTED"
  }
 ]
}
<7>19:51:48.233 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:55:08Z"
  }
 ]
}
<6>19:51:48.234 CbrsDaemon.cpp      onLoop          14145 [34;1mINF[0m Listening for 2 seconds
<7>19:51:48.234 SpvLaunchdProxy.cpp  logDBusMessage      14145 [36;1mDBG[0m
handleRequest: signal sender=:1.0 -> dest=(null) serial=156
path=/com/jmawireless/jsoft/SpvLaunchd; interface=com.jmawireless.jsoft.SpvLaunchd;
member=StartProcess; signature=s
<7>19:51:48.234 SpvLaunchdProxy.cpp  logActiveEnbs       14145 [36;1mDBG[0m Dump
activeEnbs_ map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,
"invalid_cfg":"","state":"CONNECTED"}]}
<6>19:51:49.256 CbrsDaemon.cpp      parseTree       14145 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>19:51:49.258 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:51:49.261 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:55:09Z"
  }
 ]
}
<6>19:51:49.262 ManagerEnb.cpp      command         14145 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>19:51:49.262 CbrsDaemon.cpp      onLoop          14145 [34;1mINF[0m Listening for 2 seconds
<6>19:51:49.401 Enb.cpp          onData          14175 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:51:52.286 CbrsDaemon.cpp      parseTree       14145 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:51:52.288 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:51:52.292 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "measReportConfig": [
    "RECEIVED_POWER_WITH_GRANT"
   ],
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:55:12Z"
  }
 ]
}
<6>19:51:52.292 ManagerEnb.cpp      command         14145 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>19:51:52.292 CbrsDaemon.cpp      onLoop          14145 [34;1mINF[0m Listening for 2 seconds
<6>19:51:52.392 Enb.cpp          onData          14175 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:51:55.319 CbrsDaemon.cpp      parseTree       14145 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:51:55.322 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -96
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
```

```
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3580000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3590000000,
        "measRcvdPower": -95
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3600000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3610000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3620000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3630000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3640000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3650000000,
        "measRcvdPower": -99
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3660000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3670000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3680000000,
        "measRcvdPower": -95
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3690000000,
        "measRcvdPower": -97
      }
    ]
   },
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:51:55.327 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:55:15Z"
  }
 ]
}
<6>19:51:55.327 ManagerEnb.cpp     command         14145 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>19:51:55.327 CbrsDaemon.cpp     onLoop          14145 [34;1mINF[0m Listening for 2 seconds
<6>19:51:55.427 Enb.cpp          onData          14175 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:51:58.355 CbrsDaemon.cpp     parseTree       14145 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:51:58.357 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:51:58.360 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:55:18Z"
  }
 ]
}
<6>19:51:58.360 ManagerEnb.cpp     command         14145 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>19:51:58.361 CbrsDaemon.cpp     onLoop          14145 [34;1mINF[0m Listening for 2 seconds
<6>19:51:58.461 Enb.cpp          onData          14175 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:52:01.385 CbrsDaemon.cpp     parseTree       14145 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:52:01.388 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:52:01.392 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:55:21Z"
  }
 ]
}
<6>19:52:01.392 ManagerEnb.cpp     command         14145 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>19:52:01.392 CbrsDaemon.cpp     onLoop          14145 [34;1mINF[0m Listening for 2 seconds
<6>19:52:01.492 Enb.cpp          onData          14175 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:52:04.416 CbrsDaemon.cpp     parseTree       14145 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:52:04.419 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:52:04.422 Sas.cpp          post            14145 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:55:24Z"
  }
```

```
 ]
}
<6>19:52:04.422 ManagerEnb.cpp      command          14145 [34;1mINF[0m Sending tx_expire to eNB(1),
with expiration: 60000
<6>19:52:04.423 CbrsDaemon.cpp      onLoop           14145 [34;1mINF[0m Listening for 2 seconds
<6>19:52:04.523 Enb.cpp             onData           14175 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:52:07.447 CbrsDaemon.cpp      parseTree        14145 [34;1mINF[0m Found CBRS Cell: cell_id 0,
earfcn_dl 55990
<7>19:52:07.450 Sas.cpp             post             14145 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:52:07.454 Sas.cpp             post             14145 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:55:27Z"
  }
 ]
}
<6>19:52:07.454 ManagerEnb.cpp      command          14145 [34;1mINF[0m Sending tx_expire to eNB(1),
with expiration: 60000
<6>19:52:07.455 CbrsDaemon.cpp      onLoop           14145 [34;1mINF[0m Listening for 2 seconds
<6>19:52:07.555 Enb.cpp             onData           14175 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:52:10.479 CbrsDaemon.cpp      parseTree        14145 [34;1mINF[0m Found CBRS Cell: cell_id 0,
earfcn_dl 55990
<7>19:52:10.482 Sas.cpp             post             14145 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:52:10.483 Sas.cpp             post             14145 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:55:30Z"
  }
 ]
}
<6>19:52:10.483 ManagerEnb.cpp      command          14145 [34;1mINF[0m Sending tx_expire to eNB(1),
with expiration: 60000
<6>19:52:10.484 CbrsDaemon.cpp      onLoop           14145 [34;1mINF[0m Listening for 2 seconds
<6>19:52:10.583 Enb.cpp             onData           14175 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:52:13.507 CbrsDaemon.cpp      parseTree        14145 [34;1mINF[0m Found CBRS Cell: cell_id 0,
earfcn_dl 55990
<7>19:52:13.510 Sas.cpp             post             14145 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "operationState": "AUTHORIZED"
  }
 ]
}
```

```
<7>19:52:13.511 Sas.cpp             post             14145 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "502757752",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:55:33Z"
  }
 ]
}
```

9.25 Log file for test case ID: WINNF.FT.C.RLQ.1


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT",
    "RECEIVED_POWER_WITHOUT_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>19:54:48.429 Sas.cpp post 14346 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>19:54:48.429 CbrsDaemon.cpp onLoop 14346 [34;1mINF[0m Listening for 2 seconds
<7>19:54:48.430 SpvLaunchdProxy.cpp create 14346 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>19:54:48.430 SpvLaunchdProxy.cpp create 14346 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>19:54:48.430 SpvLaunchdProxy.cpp initSpvLaunchdProxy 14346 [36;1mDBG[0m SpvLaunchd
is running.
<7>19:54:48.430 SpvLaunchdProxy.cpp logDBusMessage 14346 [36;1mDBG[0m
handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.100 serial=2
path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired;
signature=s
<7>19:54:48.430 SpvLaunchdProxy.cpp dbusHandler 14346 [36;1mDBG[0m NameAcquired:
:1.100
<7>19:54:48.430 SpvLaunchdProxy.cpp dbusHandler 14346 [36;1mDBG[0m Connection
name: :1.100
<6>19:54:49.503 CbrsDaemon.cpp parseTree 14346 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>19:54:49.505 CbrsDaemon.cpp onLoop 14346 [34;1mINF[0m Listening for 2 seconds
<6>19:54:52.530 CbrsDaemon.cpp parseTree 14346 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>19:54:52.532 CbrsDaemon.cpp onLoop 14346 [34;1mINF[0m Listening for 2 seconds
<6>19:54:55.556 CbrsDaemon.cpp parseTree 14346 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:54:55.559 CbrsDaemon.cpp persistEntities 14346 [36;1mDBG[0m Grant for cell 0,
belonging to eNB 1 created.
<6>19:54:55.560 ManagerCbsd.cpp command 14346 [34;1mINF[0m Send command to CBSD
on fe80::72b3:d5ff:fe29:c2f1:
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558979695,"user":
"user"}, with timeout of 30
<6>19:54:55.560 ManagerCbsd.cpp getResponseFromReque 14346 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Send (timeout 30 seconds):
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558979695,"user":
"user"}
<6>19:54:55.592 ManagerCbsd.cpp getResponseFromReque 14346 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Socket response received (198699 bytes)
<7>19:54:55.595 Sas.cpp post 14346 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -96
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -95
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3630000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3640000000,
      "measRcvdPower": -100
     },
```

```
      {
        "measBandwidth": 10000000,
        "measFrequency": 3650000000,
        "measRcvdPower": -99
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3660000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3670000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3680000000,
        "measRcvdPower": -94
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3690000000,
        "measRcvdPower": -97
      }
    ]
   },
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>19:54:55.602 Sas.cpp          post            14346 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T17:54:55Z",
   "grantId": "607261946",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>19:54:55.602 Sas.cpp          post            14346 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "607261946",
   "operationState": "GRANTED"
  }
 ]
}
<7>19:54:55.644 Sas.cpp          post            14346 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "607261946",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:58:15Z"
  }
 ]
}
<6>19:54:55.645 CbrsDaemon.cpp      onLoop          14346 [34;1mINF[0m Listening for 2 seconds
<7>19:54:55.645 SpvLaunchdProxy.cpp logDBusMessage  14346 [36;1mDBG[0m handleRequest:
signal sender=:1.0 -> dest=(null) serial=162 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StartProcess; signature=s
<7>19:54:55.645 SpvLaunchdProxy.cpp logActiveEnbs   14346 [36;1mDBG[0m Dump activeEnbs_
map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"in
valid_cfg":"","state":"CONNECTED"}]}
<6>19:54:56.666 CbrsDaemon.cpp      parseTree       14346 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:54:56.669 Sas.cpp          post            14346 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "607261946",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:54:56.672 Sas.cpp          post            14346 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "607261946",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:58:16Z"
  }
 ]
}
<6>19:54:56.672 ManagerEnb.cpp      command         14346 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>19:54:56.673 CbrsDaemon.cpp      onLoop          14346 [34;1mINF[0m Listening for 2 seconds
<6>19:54:56.828 Enb.cpp          onData          14377 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:54:59.697 CbrsDaemon.cpp      parseTree       14346 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:54:59.700 Sas.cpp          post            14346 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "607261946",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:54:59.703 Sas.cpp          post            14346 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "607261946",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T17:58:19Z"
  }
 ]
}
<6>19:54:59.703 ManagerEnb.cpp      command         14346 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>19:54:59.704 CbrsDaemon.cpp      onLoop          14346 [34;1mINF[0m Listening for 2 seconds
<6>19:54:59.804 Enb.cpp          onData          14377 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:55:02.728 CbrsDaemon.cpp      parseTree       14346 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:55:02.731 Sas.cpp          post            14346 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "607261946",
   "operationState": "AUTHORIZED"
  }
 ]
}
```

<7>19:55:02.734 Sas.cpp post 14346 [36;1mDBG[0m {
"heartbeatResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "607261946",
"response": {
"responseCode": 0
},
"transmitExpireTime": "2019-05-27T17:58:22Z"
}
]
}
<6>19:55:02.734 ManagerEnb.cpp command 14346 [34;1mINF[0m Sending tx_expire to eNB(1), with expiration: 60000
<6>19:55:02.735 CbrsDaemon.cpp onLoop 14346 [34;1mINF[0m Listening for 2 seconds
<6>19:55:02.834 Enb.cpp onData 14377 [34;1mINF[0m Answer received from eNB (1): flags(129), {"message":"tx_expire"}
<7>19:55:04.548 SpvLaunchdProxy.cpp logDBusMessage 14346 [36;1mDBG[0m handleRequest: signal sender=:1.0 -> dest=(null) serial=164
path=/com/jmawireless/jsoft/SpvLaunchd; interface=com.jmawireless.jsoft.SpvLaunchd; member=StopProcess; signature=s
<7>19:55:04.548 SpvLaunchdProxy.cpp logActiveEnbs 14346 [36;1mDBG[0m Dump activeEnbs_ map: {"admin_status":"UP","enbs":[]}
<6>19:55:05.572 CbrsDaemon.cpp parseTree 14346 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<7>19:55:05.575 CbrsDaemon.cpp cleanupEntities 14346 [36;1mDBG[0m All grants belonging to CBRS cell 0, eNB 1 deleted (not enabled).
<6>19:55:05.575 Grant.cpp erase 14346 [34;1mINF[0m Grant relinquishment procedure for Grant 607261946
<7>19:55:05.575 Sas.cpp post 14346 [36;1mDBG[0m {
"relinquishmentRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "607261946"
}
]
}
<7>19:55:05.579 Sas.cpp post 14346 [36;1mDBG[0m {
"relinquishmentResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "607261946",
"response": {
"responseCode": 0
}
}
]
}

9.26 Log file for test case ID: WINNF.FT.C.RLQ.3


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT",
    "RECEIVED_POWER_WITHOUT_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>19:57:34.667 Sas.cpp post 14656 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>19:57:34.667 CbrsDaemon.cpp onLoop 14656 [34;1mINF[0m Listening for 2 seconds
<7>19:57:34.668 SpvLaunchdProxy.cpp create 14656 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>19:57:34.668 SpvLaunchdProxy.cpp create 14656 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>19:57:34.668 SpvLaunchdProxy.cpp initSpvLaunchdProxy 14656 [36;1mDBG[0m SpvLaunchd
is running.
<7>19:57:34.668 SpvLaunchdProxy.cpp logDBusMessage 14656 [36;1mDBG[0m
handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.104 serial=2
path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired;
signature=s
<7>19:57:34.668 SpvLaunchdProxy.cpp dbusHandler 14656 [36;1mDBG[0m NameAcquired:
:1.104
<7>19:57:34.668 SpvLaunchdProxy.cpp dbusHandler 14656 [36;1mDBG[0m Connection
name: :1.104
<6>19:57:35.691 CbrsDaemon.cpp parseTree 14656 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>19:57:35.693 CbrsDaemon.cpp onLoop 14656 [34;1mINF[0m Listening for 2 seconds
<6>19:57:38.718 CbrsDaemon.cpp parseTree 14656 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<6>19:57:38.720 CbrsDaemon.cpp onLoop 14656 [34;1mINF[0m Listening for 2 seconds
<7>19:57:39.210 SpvLaunchdProxy.cpp logDBusMessage 14656 [36;1mDBG[0m handleRequest:
signal sender=:1.0 -> dest=(null) serial=170 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StartProcess; signature=s
<7>19:57:39.210 SpvLaunchdProxy.cpp logActiveEnbs 14656 [36;1mDBG[0m Dump activeEnbs_
map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"in
valid_cfg":"","state":"CONNECTED"}]}
<6>19:57:40.233 CbrsDaemon.cpp parseTree 14656 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:57:40.235 CbrsDaemon.cpp persistEntities 14656 [36;1mDBG[0m Grant for cell 0,
belonging to eNB 1 created.
<6>19:57:40.236 ManagerCbsd.cpp command 14656 [34;1mINF[0m Send command to CBSD
on fe80::72b3:d5ff:fe29:c2f1:
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558979860,"user":
"user"}, with timeout of 30
<6>19:57:40.236 ManagerCbsd.cpp getResponseFromReque 14656 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Send (timeout 30 seconds):
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558979860,"user":
"user"}
<6>19:57:40.269 ManagerCbsd.cpp getResponseFromReque 14656 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Socket response received (194165 bytes)
<7>19:57:40.272 Sas.cpp post 14656 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -97
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -95
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
```

```
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3630000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3640000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3650000000,
        "measRcvdPower": -87
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3660000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3670000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3680000000,
        "measRcvdPower": -95
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3690000000,
        "measRcvdPower": -97
      }
     ]
    },
    "operationParam": {
     "maxEirp": 0,
     "operationFrequencyRange": {
      "highFrequency": 3630000000,
      "lowFrequency": 3620000000
     }
    }
   }
  ]
}
<7>19:57:40.279 Sas.cpp          post            14656 [36;1mDBG[0m {
  "grantResponse": [
   {
    "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
    "channelType": "GAA",
    "grantExpireTime": "2019-06-03T17:57:40Z",
    "grantId": "10052003",
    "heartbeatInterval": 60,
    "response": {
     "responseCode": 0
    }
   }
  ]
}
<7>19:57:40.279 Sas.cpp          post            14656 [36;1mDBG[0m {
  "heartbeatRequest": [
   {
    "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
    "grantId": "10052003",
    "operationState": "GRANTED"
   }
  ]
}
<7>19:57:40.321 Sas.cpp          post            14656 [36;1mDBG[0m {
  "heartbeatResponse": [
   {
    "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
    "grantId": "10052003",
    "response": {
     "responseCode": 0
    },
    "transmitExpireTime": "2019-05-27T18:01:00Z"
   }
  ]
}
<6>19:57:40.321 ManagerEnb.cpp      command          14656 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>19:57:40.322 CbrsDaemon.cpp      onLoop           14656 [34;1mINF[0m Listening for 2 seconds
<6>19:57:41.164 Enb.cpp             onData           14691 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:57:43.346 CbrsDaemon.cpp      parseTree        14656 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:57:43.348 Sas.cpp          post            14656 [36;1mDBG[0m {
  "heartbeatRequest": [
   {
    "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
    "grantId": "10052003",
    "operationState": "AUTHORIZED"
   }
  ]
}
<7>19:57:43.352 Sas.cpp          post            14656 [36;1mDBG[0m {
  "heartbeatResponse": [
   {
    "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
    "grantId": "10052003",
    "response": {
     "responseCode": 0
    },
    "transmitExpireTime": "2019-05-27T18:01:03Z"
   }
  ]
}
<6>19:57:43.352 ManagerEnb.cpp      command          14656 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>19:57:43.352 CbrsDaemon.cpp      onLoop           14656 [34;1mINF[0m Listening for 2 seconds
<6>19:57:43.452 Enb.cpp             onData           14691 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:57:46.376 CbrsDaemon.cpp      parseTree        14656 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:57:46.379 Sas.cpp          post            14656 [36;1mDBG[0m {
  "heartbeatRequest": [
   {
    "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
    "grantId": "10052003",
    "operationState": "AUTHORIZED"
   }
  ]
}
<7>19:57:46.382 Sas.cpp          post            14656 [36;1mDBG[0m {
  "heartbeatResponse": [
   {
    "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
    "grantId": "10052003",
    "response": {
     "responseCode": 0
    },
    "transmitExpireTime": "2019-05-27T18:01:06Z"
   }
  ]
}
<6>19:57:46.382 ManagerEnb.cpp      command          14656 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>19:57:46.383 CbrsDaemon.cpp      onLoop           14656 [34;1mINF[0m Listening for 2 seconds
<6>19:57:46.483 Enb.cpp             onData           14691 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<7>19:57:47.996 SpvLaunchdProxy.cpp  logDBusMessage      14656 [36;1mDBG[0m handleRequest:
signal sender=:1.0 -> dest=(null) serial=172 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StopProcess; signature=s
<7>19:57:47.996 SpvLaunchdProxy.cpp  logActiveEnbs       14656 [36;1mDBG[0m Dump activeEnbs_
map: {"admin_status":"UP","enbs":[]}
<6>19:57:49.020 CbrsDaemon.cpp      parseTree        14656 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:57:49.023 CbrsDaemon.cpp      cleanupEntities      14656 [36;1mDBG[0m All grants belonging to
CBRS cell 0, eNB 1 deleted (not enabled).
<6>19:57:49.023 Grant.cpp           erase            14656 [34;1mINF[0m Grant relinquishment procedure
for Grant 10052003
<7>19:57:49.023 Sas.cpp          post            14656 [36;1mDBG[0m {
  "relinquishmentRequest": [
   {
    "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
    "grantId": "10052003"
   }
  ]
}
```

<7>19:57:49.027 Sas.cpp post 14656 [36;1mDBG[0m {
"relinquishmentResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"response": {
"responseCode": 102,
"responseData": [
"grantId"
]
}
}
]
}

9.27 Log file for test case ID: WINNF.FT.C.RLQ.5


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT",
    "RECEIVED_POWER_WITHOUT_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>19:59:34.826 Sas.cpp post 14844 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>19:59:34.827 CbrsDaemon.cpp onLoop 14844 [34;1mINF[0m Listening for 2 seconds
<7>19:59:34.828 SpvLaunchdProxy.cpp create 14844 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>19:59:34.828 SpvLaunchdProxy.cpp create 14844 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>19:59:34.828 SpvLaunchdProxy.cpp initSpvLaunchdProxy 14844 [36;1mDBG[0m SpvLaunchd
is running.
<7>19:59:34.828 SpvLaunchdProxy.cpp logDBusMessage 14844 [36;1mDBG[0m
handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.107 serial=2
path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired;
signature=s
<7>19:59:34.828 SpvLaunchdProxy.cpp dbusHandler 14844 [36;1mDBG[0m NameAcquired:
:1.107
<7>19:59:34.828 SpvLaunchdProxy.cpp dbusHandler 14844 [36;1mDBG[0m Connection
name: :1.107
<6>19:59:35.851 CbrsDaemon.cpp parseTree 14844 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>19:59:35.853 CbrsDaemon.cpp onLoop 14844 [34;1mINF[0m Listening for 2 seconds
<7>19:59:36.500 SpvLaunchdProxy.cpp logDBusMessage 14844 [36;1mDBG[0m handleRequest:
signal sender=:1.0 -> dest=(null) serial=177 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StartProcess; signature=s
<7>19:59:36.500 SpvLaunchdProxy.cpp logActiveEnbs 14844 [36;1mDBG[0m Dump activeEnbs_
map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"in
valid_cfg":"","state":"CONNECTED"}]}
<6>19:59:37.522 CbrsDaemon.cpp parseTree 14844 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:59:37.524 CbrsDaemon.cpp persistEntities 14844 [36;1mDBG[0m Grant for cell 0,
belonging to eNB 1 created.
<6>19:59:37.525 ManagerCbsd.cpp command 14844 [34;1mINF[0m Send command to CBSD
on fe80::72b3:d5ff:fe29:c2f1:
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558979977,"user":
"user"}, with timeout of 30
<6>19:59:37.525 ManagerCbsd.cpp getResponseFromReque 14844 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Send (timeout 30 seconds):
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558979977,"user":
"user"}
<6>19:59:37.738 ManagerCbsd.cpp getResponseFromReque 14844 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Socket response received (192427 bytes)
<7>19:59:37.742 Sas.cpp post 14844 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -99
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -96
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
     },
```

```
      {
       "measBandwidth": 10000000,
       "measFrequency": 3630000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3640000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3650000000,
       "measRcvdPower": -99
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3660000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3670000000,
       "measRcvdPower": -100
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3680000000,
       "measRcvdPower": -96
      },
      {
       "measBandwidth": 10000000,
       "measFrequency": 3690000000,
       "measRcvdPower": -98
      }
     ]
    },
    "operationParam": {
     "maxEirp": 0,
     "operationFrequencyRange": {
      "highFrequency": 3630000000,
      "lowFrequency": 3620000000
     }
    }
   }
 ]
}
<7>19:59:37.749 Sas.cpp          post            14844 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T17:59:37Z",
   "grantId": "844049033",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>19:59:37.749 Sas.cpp          post            14844 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "844049033",
   "operationState": "GRANTED"
  }
 ]
}
<7>19:59:37.791 Sas.cpp          post            14844 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "844049033",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:02:57Z"
  }
 ]
}
<6>19:59:37.791 ManagerEnb.cpp     command         14844 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>19:59:37.792 CbrsDaemon.cpp     onLoop          14844 [34;1mINF[0m Listening for 2 seconds
<6>19:59:38.039 Enb.cpp            onData          14879 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:59:40.816 CbrsDaemon.cpp     parseTree       14844 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:59:40.818 Sas.cpp          post            14844 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "844049033",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:59:40.822 Sas.cpp          post            14844 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "844049033",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:03:00Z"
  }
 ]
}
<6>19:59:40.822 ManagerEnb.cpp     command         14844 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>19:59:40.822 CbrsDaemon.cpp     onLoop          14844 [34;1mINF[0m Listening for 2 seconds
<6>19:59:40.922 Enb.cpp            onData          14879 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>19:59:43.846 CbrsDaemon.cpp     parseTree       14844 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:59:43.849 Sas.cpp          post            14844 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "844049033",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>19:59:43.852 Sas.cpp          post            14844 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "844049033",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:03:03Z"
  }
 ]
}
<6>19:59:43.852 ManagerEnb.cpp     command         14844 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>19:59:43.853 CbrsDaemon.cpp     onLoop          14844 [34;1mINF[0m Listening for 2 seconds
<6>19:59:43.952 Enb.cpp            onData          14879 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<7>19:59:45.064 SpvLaunchdProxy.cpp logDBusMessage     14844 [36;1mDBG[0m handleRequest:
signal sender=:1.0 -> dest=(null) serial=179 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StopProcess; signature=s
<7>19:59:45.064 SpvLaunchdProxy.cpp logActiveEnbs      14844 [36;1mDBG[0m Dump activeEnbs_
map: {"admin_status":"UP","enbs":[]}
<6>19:59:46.087 CbrsDaemon.cpp     parseTree       14844 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>19:59:46.090 CbrsDaemon.cpp     cleanupEntities     14844 [36;1mDBG[0m All grants belonging to
CBRS cell 0, eNB 1 deleted (not enabled).
<6>19:59:46.090 Grant.cpp          erase           14844 [34;1mINF[0m Grant relinquishment procedure
for Grant 844049033
<7>19:59:46.090 Sas.cpp          post            14844 [36;1mDBG[0m {
 "relinquishmentRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "844049033"
  }
 ]
}
```

<7>19:59:46.093 Sas.cpp post 14844 [36;1mDBG[0m {
"relinquishmentResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"response": {
"responseCode": 103,
"responseData": [
"grantId"
]
}
}
]
}

9.28 Log file for test case ID: WINNF.FT.C.DRG.1


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT",
    "RECEIVED_POWER_WITHOUT_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>20:03:06.280 Sas.cpp post 15074 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>20:03:06.281 CbrsDaemon.cpp onLoop 15074 [34;1mINF[0m Listening for 2 seconds
<7>20:03:06.281 SpvLaunchdProxy.cpp create 15074 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>20:03:06.281 SpvLaunchdProxy.cpp create 15074 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>20:03:06.282 SpvLaunchdProxy.cpp initSpvLaunchdProxy 15074 [36;1mDBG[0m SpvLaunchd
is running.
<7>20:03:06.282 SpvLaunchdProxy.cpp logDBusMessage 15074 [36;1mDBG[0m
handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.109 serial=2
path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired;
signature=s
<7>20:03:06.282 SpvLaunchdProxy.cpp dbusHandler 15074 [36;1mDBG[0m NameAcquired:
:1.109
<7>20:03:06.282 SpvLaunchdProxy.cpp dbusHandler 15074 [36;1mDBG[0m Connection
name: :1.109
<6>20:03:07.304 CbrsDaemon.cpp parseTree 15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:03:07.306 CbrsDaemon.cpp onLoop 15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:10.331 CbrsDaemon.cpp parseTree 15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:03:10.333 CbrsDaemon.cpp onLoop 15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:13.358 CbrsDaemon.cpp parseTree 15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:03:13.360 CbrsDaemon.cpp onLoop 15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:16.384 CbrsDaemon.cpp parseTree 15074 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>20:03:16.388 CbrsDaemon.cpp persistEntities 15074 [36;1mDBG[0m Grant for cell 0,
belonging to eNB 1 created.
<6>20:03:16.388 ManagerCbsd.cpp command 15074 [34;1mINF[0m Send command to CBSD
on fe80::72b3:d5ff:fe29:c2f1:
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558980196,"user":
"user"}, with timeout of 30
<6>20:03:16.388 ManagerCbsd.cpp getResponseFromReque 15074 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Send (timeout 30 seconds):
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558980196,"user":
"user"}
<6>20:03:16.420 ManagerCbsd.cpp getResponseFromReque 15074 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Socket response received (186146 bytes)
<7>20:03:16.423 Sas.cpp post 15074 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -98
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -89
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3630000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3640000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3650000000,
      "measRcvdPower": -82
     },
     {
```

```
        "measBandwidth": 10000000,
        "measFrequency": 3660000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3670000000,
        "measRcvdPower": -100
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3680000000,
        "measRcvdPower": -96
       },
       {
        "measBandwidth": 10000000,
        "measFrequency": 3690000000,
        "measRcvdPower": -97
       }
      ]
     },
     "operationParam": {
      "maxEirp": 0,
      "operationFrequencyRange": {
       "highFrequency": 3630000000,
       "lowFrequency": 3620000000
      }
     }
    }
   ]
}
<7>20:03:16.430 Sas.cpp          post            15074 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T18:03:16Z",
   "grantId": "873620434",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>20:03:16.430 Sas.cpp          post            15074 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "873620434",
   "operationState": "GRANTED"
  }
 ]
}
<7>20:03:16.473 Sas.cpp          post            15074 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "873620434",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:06:36Z"
  }
 ]
}
<6>20:03:16.474 CbrsDaemon.cpp      onLoop          15074 [34;1mINF[0m Listening for 2 seconds
<7>20:03:16.474 SpvLaunchdProxy.cpp  logDBusMessage     15074 [36;1mDBG[0m
handleRequest: signal sender=:1.0 -> dest=(null) serial=183
path=/com/jmawireless/jsoft/SpvLaunchd; interface=com.jmawireless.jsoft.SpvLaunchd;
member=StartProcess; signature=s
<7>20:03:16.474 SpvLaunchdProxy.cpp  logActiveEnbs      15074 [36;1mDBG[0m Dump
activeEnbs_ map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,
"invalid_cfg":"","state":"CONNECTED"}]}
<6>20:03:17.496 CbrsDaemon.cpp      parseTree       15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>20:03:17.498 Sas.cpp          post            15074 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "873620434",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>20:03:17.514 Sas.cpp          post            15074 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "873620434",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:06:37Z"
  }
 ]
}
<6>20:03:17.514 ManagerEnb.cpp      command         15074 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>20:03:17.519 CbrsDaemon.cpp      onLoop          15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:18.061 Enb.cpp          onData          15107 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>20:03:20.543 CbrsDaemon.cpp      parseTree       15074 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>20:03:20.545 Sas.cpp          post            15074 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "873620434",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>20:03:20.548 Sas.cpp          post            15074 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "873620434",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:06:40Z"
  }
 ]
}
<6>20:03:20.549 ManagerEnb.cpp      command         15074 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>20:03:20.549 CbrsDaemon.cpp      onLoop          15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:20.649 Enb.cpp          onData          15107 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>20:03:23.576 CbrsDaemon.cpp      parseTree       15074 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>20:03:23.578 Sas.cpp          post            15074 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "873620434",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>20:03:23.581 Sas.cpp          post            15074 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "873620434",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:06:43Z"
  }
 ]
}
```

```
<6>20:03:23.581 ManagerEnb.cpp      command          15074 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>20:03:23.582 CbrsDaemon.cpp      onLoop           15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:23.682 Enb.cpp             onData           15107 [34;1mINF[0m Answer received from eNB
(1): flags(129), {"message":"tx_expire"}
<7>20:03:25.194 SpvLaunchdProxy.cpp  logDBusMessage      15074 [36;1mDBG[0m
handleRequest: signal sender=:1.0 -> dest=(null) serial=185
path=/com/jmawireless/jsoft/SpvLaunchd; interface=com.jmawireless.jsoft.SpvLaunchd;
member=StopProcess; signature=s
<7>20:03:25.194 SpvLaunchdProxy.cpp  logActiveEnbs       15074 [36;1mDBG[0m Dump
activeEnbs_ map: {"admin_status":"UP","enbs":[]}
<6>20:03:26.218 CbrsDaemon.cpp      parseTree        15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<7>20:03:26.220 CbrsDaemon.cpp      cleanupEntities      15074 [36;1mDBG[0m All grants
belonging to CBRS cell 0, eNB 1 deleted (not enabled).
<6>20:03:26.221 Grant.cpp           erase            15074 [34;1mINF[0m Grant relinquishment
procedure for Grant 873620434
<7>20:03:26.221 Sas.cpp             post             15074 [36;1mDBG[0m {
 "relinquishmentRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "873620434"
  }
 ]
}
<7>20:03:26.224 Sas.cpp             post             15074 [36;1mDBG[0m {
 "relinquishmentResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "873620434",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>20:03:26.225 CbrsDaemon.cpp      onLoop           15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:29.249 CbrsDaemon.cpp      parseTree        15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:03:29.252 CbrsDaemon.cpp      onLoop           15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:32.277 CbrsDaemon.cpp      parseTree        15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:03:32.280 CbrsDaemon.cpp      onLoop           15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:35.305 CbrsDaemon.cpp      parseTree        15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:03:35.307 CbrsDaemon.cpp      onLoop           15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:38.331 CbrsDaemon.cpp      parseTree        15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:03:38.334 CbrsDaemon.cpp      onLoop           15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:41.359 CbrsDaemon.cpp      parseTree        15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:03:41.362 CbrsDaemon.cpp      onLoop           15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:44.386 CbrsDaemon.cpp      parseTree        15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:03:44.388 CbrsDaemon.cpp      onLoop           15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:47.413 CbrsDaemon.cpp      parseTree        15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:03:47.416 CbrsDaemon.cpp      onLoop           15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:50.441 CbrsDaemon.cpp      parseTree        15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:03:50.444 CbrsDaemon.cpp      onLoop           15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:53.469 CbrsDaemon.cpp      parseTree        15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:03:53.471 CbrsDaemon.cpp      onLoop           15074 [34;1mINF[0m Listening for 2 seconds
<6>20:03:56.489 CbrsDaemon.cpp      parseTree        15074 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:03:56.489 Cbsd.cpp            erase            15074 [34;1mINF[0m Deregistration procedure for
CBSD XM2-X19AX35M2Mock-SAS1012482003
<7>20:03:56.489 Sas.cpp             post             15074 [36;1mDBG[0m {
 "deregistrationRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003"
  }
 ]
}
<7>20:03:56.492 Sas.cpp             post             15074 [36;1mDBG[0m {
 "deregistrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
```

9.29 Log file for test case ID: WINNF.FT.C.DRG.3


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT",
    "RECEIVED_POWER_WITHOUT_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>20:06:22.835 Sas.cpp            post              15265 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>20:06:22.836 CbrsDaemon.cpp      onLoop            15265 [34;1mINF[0m Listening for 2 seconds
<7>20:06:22.837 SpvLaunchdProxy.cpp  create            15265 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>20:06:22.837 SpvLaunchdProxy.cpp  create            15265 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>20:06:22.837 SpvLaunchdProxy.cpp  initSpvLaunchdProxy  15265 [36;1mDBG[0m SpvLaunchd
is running.
<7>20:06:22.837 SpvLaunchdProxy.cpp  logDBusMessage      15265 [36;1mDBG[0m
handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.111 serial=2
path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired;
signature=s
<7>20:06:22.837 SpvLaunchdProxy.cpp  dbusHandler        15265 [36;1mDBG[0m NameAcquired:
:1.111
<7>20:06:22.837 SpvLaunchdProxy.cpp  dbusHandler        15265 [36;1mDBG[0m Connection
name: :1.111
<6>20:06:23.859 CbrsDaemon.cpp      parseTree         15265 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:06:23.862 CbrsDaemon.cpp      onLoop            15265 [34;1mINF[0m Listening for 2 seconds
<6>20:06:26.886 CbrsDaemon.cpp      parseTree         15265 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:06:26.888 CbrsDaemon.cpp      onLoop            15265 [34;1mINF[0m Listening for 2 seconds
<7>20:06:27.269 SpvLaunchdProxy.cpp  logDBusMessage      15265 [36;1mDBG[0m handleRequest:
signal sender=:1.0 -> dest=(null) serial=189 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StartProcess; signature=s
<7>20:06:27.269 SpvLaunchdProxy.cpp  logActiveEnbs       15265 [36;1mDBG[0m Dump activeEnbs_
map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"in
valid_cfg":"","state":"CONNECTED"}]}
<6>20:06:28.291 CbrsDaemon.cpp      parseTree         15265 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>20:06:28.293 CbrsDaemon.cpp      persistEntities     15265 [36;1mDBG[0m Grant for cell 0,
belonging to eNB 1 created.
<6>20:06:28.294 ManagerCbsd.cpp     command           15265 [34;1mINF[0m Send command to CBSD
on fe80::72b3:d5ff:fe29:c2f1:
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558980388,"user":
"user"}, with timeout of 30
<6>20:06:28.294 ManagerCbsd.cpp     getResponseFromReque 15265 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Send (timeout 30 seconds):
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558980388,"user":
"user"}
<6>20:06:28.364 ManagerCbsd.cpp     getResponseFromReque 15265 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Socket response received (183523 bytes)
<7>20:06:28.368 Sas.cpp            post              15265 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -98
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -96
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3630000000,
      "measRcvdPower": -100
     },
```

```
      {
        "measBandwidth": 10000000,
        "measFrequency": 3640000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3650000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3660000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3670000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3680000000,
        "measRcvdPower": -96
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3690000000,
        "measRcvdPower": -99
      }
     ]
    },
    "operationParam": {
     "maxEirp": 0,
     "operationFrequencyRange": {
      "highFrequency": 3630000000,
      "lowFrequency": 3620000000
     }
    }
   }
  ]
}
<7>20:06:28.374 Sas.cpp          post            15265 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T18:06:28Z",
   "grantId": "141302727",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>20:06:28.374 Sas.cpp          post            15265 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "141302727",
   "operationState": "GRANTED"
  }
 ]
}
<7>20:06:28.417 Sas.cpp          post            15265 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "141302727",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:09:48Z"
  }
 ]
}
<6>20:06:28.417 ManagerEnb.cpp      command         15265 [34;1mINF[0m Sending tx_expire to eNB(1), with
expiration: 60000
<6>20:06:28.418 CbrsDaemon.cpp      onLoop          15265 [34;1mINF[0m Listening for 2 seconds
<6>20:06:29.066 Enb.cpp             onData          15296 [34;1mINF[0m Answer received from eNB (1): flags(129),
{"message":"tx_expire"}
<6>20:06:31.442 CbrsDaemon.cpp      parseTree       15265 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl
55990
<7>20:06:31.445 Sas.cpp          post            15265 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "141302727",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>20:06:31.448 Sas.cpp          post            15265 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "141302727",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:09:51Z"
  }
 ]
}
<6>20:06:31.448 ManagerEnb.cpp      command         15265 [34;1mINF[0m Sending tx_expire to eNB(1), with
expiration: 60000
<6>20:06:31.449 CbrsDaemon.cpp      onLoop          15265 [34;1mINF[0m Listening for 2 seconds
<6>20:06:31.548 Enb.cpp             onData          15296 [34;1mINF[0m Answer received from eNB (1): flags(129),
{"message":"tx_expire"}
<6>20:06:34.476 CbrsDaemon.cpp      parseTree       15265 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl
55990
<7>20:06:34.478 Sas.cpp          post            15265 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "141302727",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>20:06:34.481 Sas.cpp          post            15265 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "141302727",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:09:54Z"
  }
 ]
}
<6>20:06:34.481 ManagerEnb.cpp      command         15265 [34;1mINF[0m Sending tx_expire to eNB(1), with
expiration: 60000
<6>20:06:34.482 CbrsDaemon.cpp      onLoop          15265 [34;1mINF[0m Listening for 2 seconds
<6>20:06:34.582 Enb.cpp             onData          15296 [34;1mINF[0m Answer received from eNB (1): flags(129),
{"message":"tx_expire"}
<7>20:06:35.091 SpvLaunchdProxy.cpp  logDBusMessage     15265 [36;1mDBG[0m handleRequest: signal sender=:1.0
-> dest=(null) serial=191 path=/com/jmawireless/jsoft/SpvLaunchd; interface=com.jmawireless.jsoft.SpvLaunchd;
member=StopProcess; signature=s
<7>20:06:35.091 SpvLaunchdProxy.cpp  logActiveEnbs      15265 [36;1mDBG[0m Dump activeEnbs_ map:
{"admin_status":"UP","enbs":[]}
<6>20:06:36.114 CbrsDaemon.cpp      parseTree       15265 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl
55990
<7>20:06:36.117 CbrsDaemon.cpp      cleanupEntities     15265 [36;1mDBG[0m All grants belonging to CBRS cell 0, eNB
1 deleted (not enabled).
<6>20:06:36.117 Grant.cpp           erase           15265 [34;1mINF[0m Grant relinquishment procedure for Grant
141302727
<7>20:06:36.117 Sas.cpp          post            15265 [36;1mDBG[0m {
 "relinquishmentRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "141302727"
  }
 ]
}
```

<7>20:06:36.120 Sas.cpp post 15265 [36;1mDBG[0m {
"relinquishmentResponse": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
"grantId": "141302727",
"response": {
"responseCode": 0
}
}
]
}
<6>20:06:36.121 CbrsDaemon.cpp onLoop 15265 [34;1mINF[0m Listening for 2 seconds
<6>20:06:39.146 CbrsDaemon.cpp parseTree 15265 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:06:39.149 CbrsDaemon.cpp onLoop 15265 [34;1mINF[0m Listening for 2 seconds
<6>20:06:42.173 CbrsDaemon.cpp parseTree 15265 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:06:42.176 CbrsDaemon.cpp onLoop 15265 [34;1mINF[0m Listening for 2 seconds
<6>20:06:45.201 CbrsDaemon.cpp parseTree 15265 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:06:45.204 CbrsDaemon.cpp onLoop 15265 [34;1mINF[0m Listening for 2 seconds
<6>20:06:48.228 CbrsDaemon.cpp parseTree 15265 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:06:48.231 CbrsDaemon.cpp onLoop 15265 [34;1mINF[0m Listening for 2 seconds
<6>20:06:51.256 CbrsDaemon.cpp parseTree 15265 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:06:51.259 CbrsDaemon.cpp onLoop 15265 [34;1mINF[0m Listening for 2 seconds
<6>20:06:54.284 CbrsDaemon.cpp parseTree 15265 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:06:54.287 CbrsDaemon.cpp onLoop 15265 [34;1mINF[0m Listening for 2 seconds
<6>20:06:57.311 CbrsDaemon.cpp parseTree 15265 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:06:57.314 CbrsDaemon.cpp onLoop 15265 [34;1mINF[0m Listening for 2 seconds
<6>20:07:00.339 CbrsDaemon.cpp parseTree 15265 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:07:00.342 CbrsDaemon.cpp onLoop 15265 [34;1mINF[0m Listening for 2 seconds
<6>20:07:03.358 CbrsDaemon.cpp parseTree 15265 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:07:03.358 Cbsd.cpp erase 15265 [34;1mINF[0m Deregistration procedure for CBSD XM2-X19AX35M2Mock-SAS1012482003
<7>20:07:03.358 Sas.cpp post 15265 [36;1mDBG[0m {
"deregistrationRequest": [
{
"cbsdId": "XM2-X19AX35M2Mock-SAS1012482003"
}
]
}
<7>20:07:03.362 Sas.cpp post 15265 [36;1mDBG[0m {
"deregistrationResponse": [
{
"response": {
"responseCode": 102
}
}
]
}

9.30 Log file for test case ID: WINNF.FT.C.DRG.5


```
{
 "registrationRequest": [
  {
   "airInterface": {
    "radioTechnology": "E_UTRA"
   },
   "callSign": "?",
   "cbsdCategory": "A",
   "cbsdInfo": {
    "firmwareVersion": "v2.0.5",
    "hardwareVersion": "v1.0.45",
    "model": "CPRI_DEVICE-XXX",
    "softwareVersion": "v1.2.1",
    "vendor": "JMA Wireless"
   },
   "cbsdSerialNumber": "1012482003",
   "fccId": "XM2-X19AX35M2",
   "installationParam": {
    "antennaAzimuth": 70,
    "antennaBeamwidth": 45,
    "antennaDowntilt": 36,
    "antennaGain": 0,
    "antennaModel": "CPRI_DEVICE-XXX-ext-antenna",
    "eirpCapability": 15,
    "height": 15.0,
    "heightType": "AMSL",
    "horizontalAccuracy": 49,
    "indoorDeployment": true,
    "latitude": 43.09,
    "longitude": -76.15,
    "verticalAccuracy": 2
   },
   "measCapability": [
    "RECEIVED_POWER_WITH_GRANT",
    "RECEIVED_POWER_WITHOUT_GRANT"
   ],
   "userId": "abc"
  }
 ]
}
<7>20:08:54.475 Sas.cpp post 15447 [36;1mDBG[0m {
 "registrationResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>20:08:54.476 CbrsDaemon.cpp onLoop 15447 [34;1mINF[0m Listening for 2 seconds
<7>20:08:54.476 SpvLaunchdProxy.cpp create 15447 [36;1mDBG[0m Added match-rule:
"sender='com.jmawireless.jsoft.SpvLaunchd',interface='com.jmawireless.jsoft.SpvLaunchd'"
<7>20:08:54.476 SpvLaunchdProxy.cpp create 15447 [36;1mDBG[0m Added match-rule:
"sender='org.freedesktop.DBus',interface='org.freedesktop.DBus'"
<7>20:08:54.477 SpvLaunchdProxy.cpp initSpvLaunchdProxy 15447 [36;1mDBG[0m SpvLaunchd
is running.
<7>20:08:54.477 SpvLaunchdProxy.cpp logDBusMessage 15447 [36;1mDBG[0m
handleRequest: signal sender=org.freedesktop.DBus -> dest=:1.113 serial=2
path=/org/freedesktop/DBus; interface=org.freedesktop.DBus; member=NameAcquired;
signature=s
<7>20:08:54.477 SpvLaunchdProxy.cpp dbusHandler 15447 [36;1mDBG[0m NameAcquired:
:1.113
<7>20:08:54.477 SpvLaunchdProxy.cpp dbusHandler 15447 [36;1mDBG[0m Connection
name: :1.113
<6>20:08:55.499 CbrsDaemon.cpp parseTree 15447 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:08:55.502 CbrsDaemon.cpp onLoop 15447 [34;1mINF[0m Listening for 2 seconds
<6>20:08:58.525 CbrsDaemon.cpp parseTree 15447 [34;1mINF[0m Found CBRS Cell:
cell_id 0, earfcn_dl 55990
<6>20:08:58.527 CbrsDaemon.cpp onLoop 15447 [34;1mINF[0m Listening for 2 seconds
<6>20:09:01.554 CbrsDaemon.cpp parseTree 15447 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>20:09:01.556 CbrsDaemon.cpp persistEntities 15447 [36;1mDBG[0m Grant for cell 0,
belonging to eNB 1 created.
<6>20:09:01.557 ManagerCbsd.cpp command 15447 [34;1mINF[0m Send command to CBSD
on fe80::72b3:d5ff:fe29:c2f1:
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558980541,"user":
"user"}, with timeout of 30
<6>20:09:01.557 ManagerCbsd.cpp getResponseFromReque 15447 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Send (timeout 30 seconds):
{"attributes":{},"operation":"get","path":"/power_vectors","type":"request","uid":1558980541,"user":
"user"}
<6>20:09:01.589 ManagerCbsd.cpp getResponseFromReque 15447 [34;1mINF[0m
[fe80::72b3:d5ff:fe29:c2f1 : 5556] Socket response received (183554 bytes)
<7>20:09:01.592 Sas.cpp post 15447 [36;1mDBG[0m {
 "grantRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "measReport": {
    "rcvdPowerMeasReports": [
     {
      "measBandwidth": 10000000,
      "measFrequency": 3550000000,
      "measRcvdPower": -99
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3560000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3570000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3580000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3590000000,
      "measRcvdPower": -86
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3600000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3610000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3620000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3630000000,
      "measRcvdPower": -100
     },
     {
      "measBandwidth": 10000000,
      "measFrequency": 3640000000,
      "measRcvdPower": -100
     },
```

```
      {
        "measBandwidth": 10000000,
        "measFrequency": 3650000000,
        "measRcvdPower": -83
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3660000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3670000000,
        "measRcvdPower": -100
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3680000000,
        "measRcvdPower": -95
      },
      {
        "measBandwidth": 10000000,
        "measFrequency": 3690000000,
        "measRcvdPower": -98
      }
    ]
   },
   "operationParam": {
    "maxEirp": 0,
    "operationFrequencyRange": {
     "highFrequency": 3630000000,
     "lowFrequency": 3620000000
    }
   }
  }
 ]
}
<7>20:09:01.599 Sas.cpp          post            15447 [36;1mDBG[0m {
 "grantResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "channelType": "GAA",
   "grantExpireTime": "2019-06-03T18:09:01Z",
   "grantId": "238753793",
   "heartbeatInterval": 60,
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<7>20:09:01.599 Sas.cpp          post            15447 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "238753793",
   "operationState": "GRANTED"
  }
 ]
}
<7>20:09:01.641 Sas.cpp          post            15447 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "238753793",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:12:21Z"
  }
 ]
}
<6>20:09:01.642 CbrsDaemon.cpp      onLoop          15447 [34;1mINF[0m Listening for 2 seconds
<7>20:09:01.642 SpvLaunchdProxy.cpp  logDBusMessage     15447 [36;1mDBG[0m handleRequest:
signal sender=:1.0 -> dest=(null) serial=195 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StartProcess; signature=s
<7>20:09:01.642 SpvLaunchdProxy.cpp  logActiveEnbs       15447 [36;1mDBG[0m Dump activeEnbs_
map:
{"admin_status":"UP","enbs":[{"cell_status":[{"cell_id":0,"cell_key":1,"locked":false}],"enb_key":1,"in
valid_cfg":"","state":"CONNECTED"}]}
<6>20:09:02.667 CbrsDaemon.cpp      parseTree       15447 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>20:09:02.669 Sas.cpp          post            15447 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "238753793",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>20:09:02.673 Sas.cpp          post            15447 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "238753793",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:12:22Z"
  }
 ]
}
<6>20:09:02.673 ManagerEnb.cpp      command         15447 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>20:09:02.673 CbrsDaemon.cpp      onLoop          15447 [34;1mINF[0m Listening for 2 seconds
<6>20:09:02.812 Enb.cpp          onData          15478 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<6>20:09:05.701 CbrsDaemon.cpp      parseTree       15447 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
<7>20:09:05.704 Sas.cpp          post            15447 [36;1mDBG[0m {
 "heartbeatRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "238753793",
   "operationState": "AUTHORIZED"
  }
 ]
}
<7>20:09:05.707 Sas.cpp          post            15447 [36;1mDBG[0m {
 "heartbeatResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "238753793",
   "response": {
    "responseCode": 0
   },
   "transmitExpireTime": "2019-05-27T18:12:25Z"
  }
 ]
}
<6>20:09:05.707 ManagerEnb.cpp      command         15447 [34;1mINF[0m Sending tx_expire to
eNB(1), with expiration: 60000
<6>20:09:05.707 CbrsDaemon.cpp      onLoop          15447 [34;1mINF[0m Listening for 2 seconds
<6>20:09:05.807 Enb.cpp          onData          15478 [34;1mINF[0m Answer received from eNB (1):
flags(129), {"message":"tx_expire"}
<7>20:09:07.622 SpvLaunchdProxy.cpp  logDBusMessage     15447 [36;1mDBG[0m handleRequest:
signal sender=:1.0 -> dest=(null) serial=197 path=/com/jmawireless/jsoft/SpvLaunchd;
interface=com.jmawireless.jsoft.SpvLaunchd; member=StopProcess; signature=s
<7>20:09:07.622 SpvLaunchdProxy.cpp  logActiveEnbs       15447 [36;1mDBG[0m Dump activeEnbs_
map: {"admin_status":"UP","enbs":[]}
<6>20:09:08.646 CbrsDaemon.cpp      parseTree       15447 [34;1mINF[0m Found CBRS Cell: cell_id
0, earfcn_dl 55990
```

```
<7>20:09:08.649 CbrsDaemon.cpp      cleanupEntities     15447 [36;1mDBG[0m All grants belonging to CBRS cell 0, eNB 1 deleted (not enabled).
<6>20:09:08.649 Grant.cpp           erase               15447 [34;1mINF[0m Grant relinquishment procedure for Grant 238753793
<7>20:09:08.649 Sas.cpp             post                15447 [36;1mDBG[0m {
 "relinquishmentRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "238753793"
  }
 ]
}
<7>20:09:08.652 Sas.cpp             post                15447 [36;1mDBG[0m {
 "relinquishmentResponse": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003",
   "grantId": "238753793",
   "response": {
    "responseCode": 0
   }
  }
 ]
}
<6>20:09:08.653 CbrsDaemon.cpp      onLoop              15447 [34;1mINF[0m Listening for 2 seconds
<6>20:09:11.679 CbrsDaemon.cpp      parseTree           15447 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:09:11.682 CbrsDaemon.cpp      onLoop              15447 [34;1mINF[0m Listening for 2 seconds
<6>20:09:14.707 CbrsDaemon.cpp      parseTree           15447 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:09:14.710 CbrsDaemon.cpp      onLoop              15447 [34;1mINF[0m Listening for 2 seconds
<6>20:09:17.733 CbrsDaemon.cpp      parseTree           15447 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:09:17.737 CbrsDaemon.cpp      onLoop              15447 [34;1mINF[0m Listening for 2 seconds
<6>20:09:20.762 CbrsDaemon.cpp      parseTree           15447 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:09:20.765 CbrsDaemon.cpp      onLoop              15447 [34;1mINF[0m Listening for 2 seconds
<6>20:09:23.789 CbrsDaemon.cpp      parseTree           15447 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:09:23.793 CbrsDaemon.cpp      onLoop              15447 [34;1mINF[0m Listening for 2 seconds
<6>20:09:26.818 CbrsDaemon.cpp      parseTree           15447 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:09:26.821 CbrsDaemon.cpp      onLoop              15447 [34;1mINF[0m Listening for 2 seconds
<6>20:09:29.844 CbrsDaemon.cpp      parseTree           15447 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:09:29.847 CbrsDaemon.cpp      onLoop              15447 [34;1mINF[0m Listening for 2 seconds
<6>20:09:32.865 CbrsDaemon.cpp      parseTree           15447 [34;1mINF[0m Found CBRS Cell: cell_id 0, earfcn_dl 55990
<6>20:09:32.866 Cbsd.cpp            erase               15447 [34;1mINF[0m Deregistration procedure for CBSD XM2-X19AX35M2Mock-SAS1012482003
<7>20:09:32.866 Sas.cpp             post                15447 [36;1mDBG[0m {
 "deregistrationRequest": [
  {
   "cbsdId": "XM2-X19AX35M2Mock-SAS1012482003"
  }
 ]
}
<7>20:09:32.869 Sas.cpp             post                15447 [36;1mDBG[0m {
 "deregistrationResponse": [
  {
   "response": {
    "responseCode": 103,
    "responseData": [
     "cbsdId"
    ]
   }
  }
 ]
}
```

9.31 Wireshark capture screenshot for test case ID: WINNF.FT.C.SCS.1

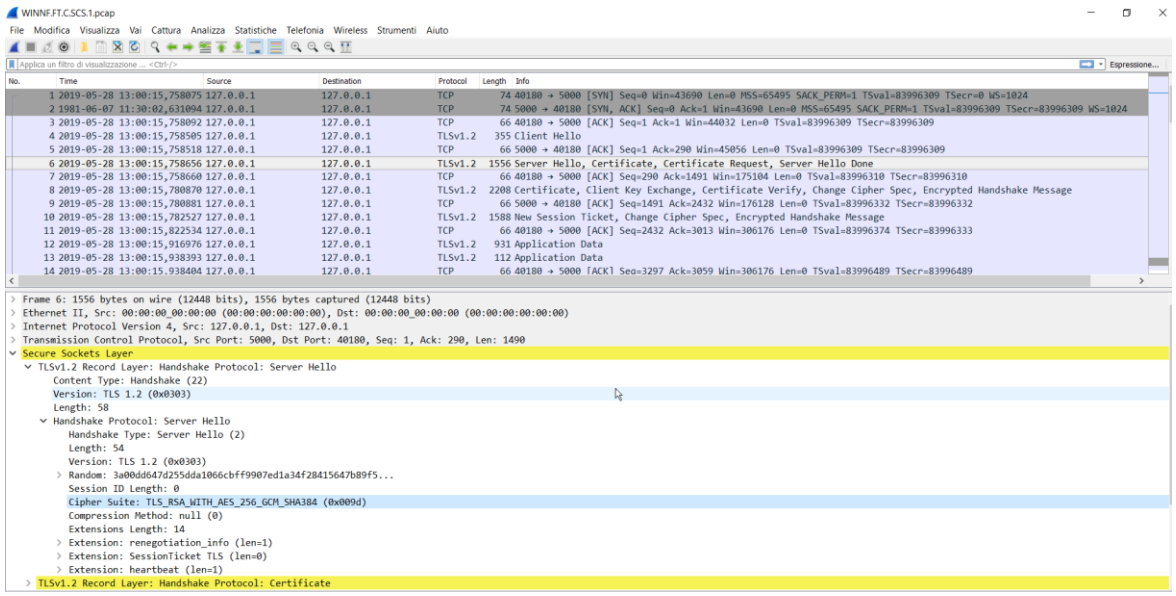

Figure 9.31-1: *Client hello*

9.32 Wireshark capture screenshot for test case ID: WINNF.FT.C.SCS.2

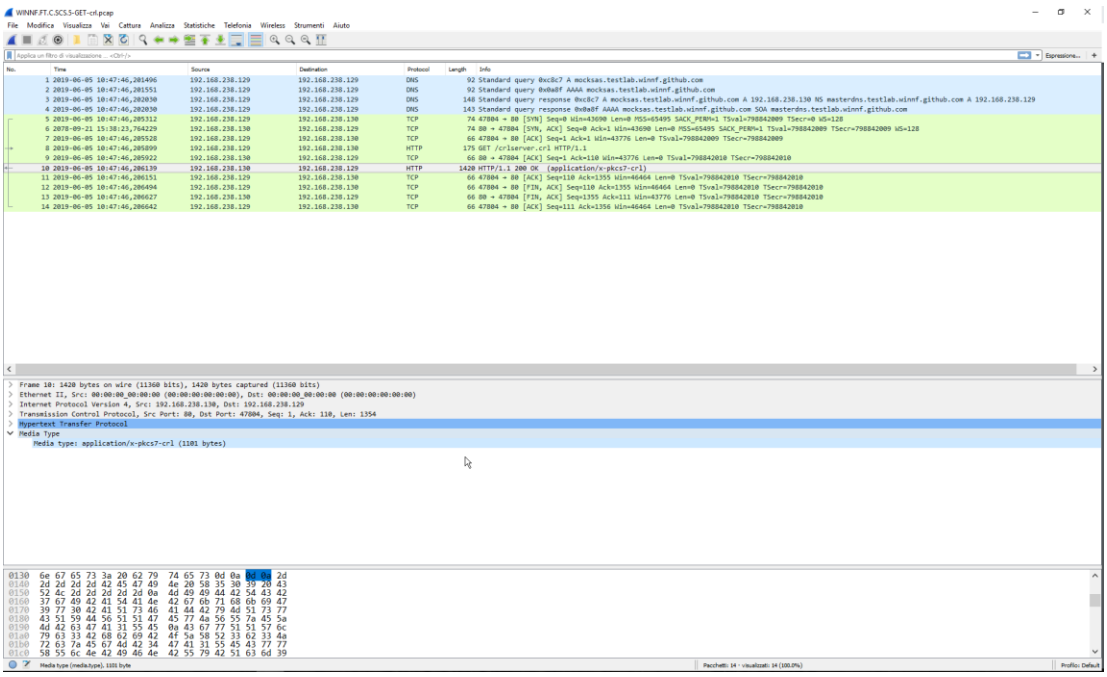

Figure 9.32-1: *DNS, GET CRL file*

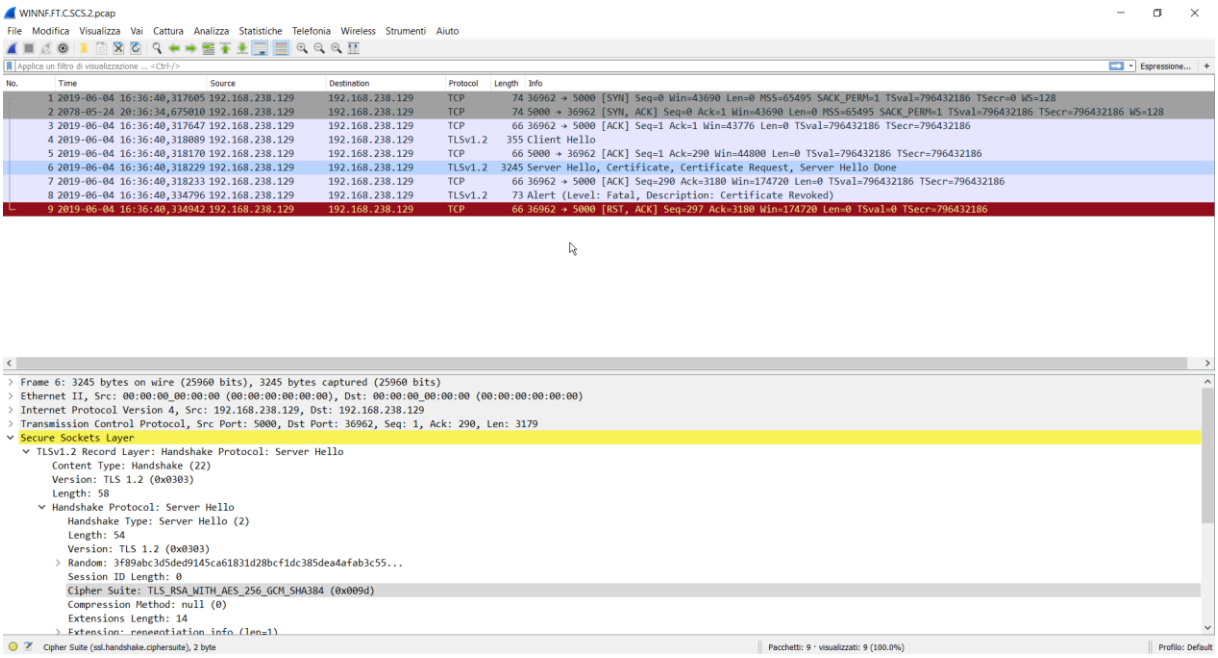

Figure 9.32-2: *Client hello, Server hello, Alert Certificate Revoked*

9.33 Wireshark capture screenshot for test case ID: WINNF.FT.C.SCS.3

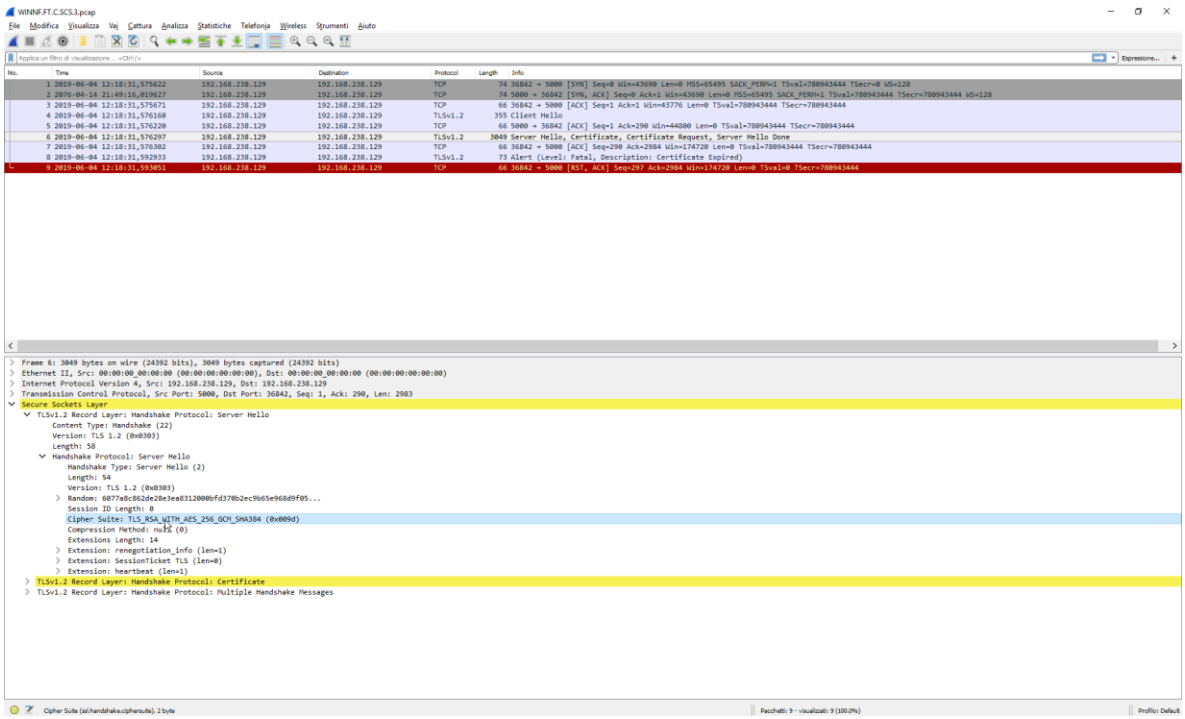

Figure 9.33-1: *Client hello, Server hello, Alert Certificate Expired*

9.34 Wireshark capture screenshot for test case ID: WINNF.FT.C.SCS.4

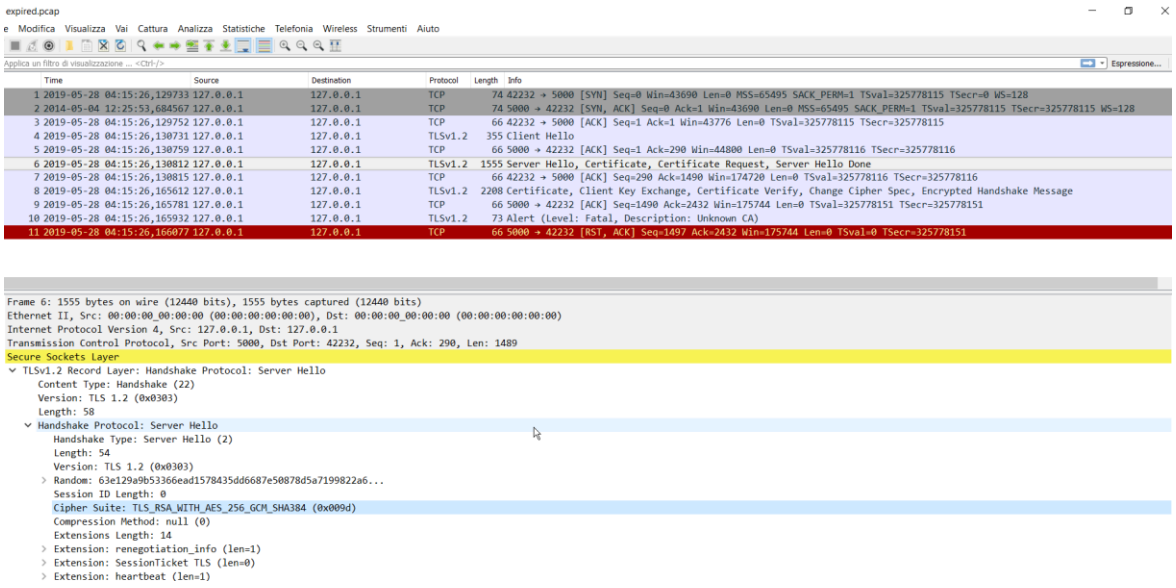

Figure 9.34-1: *Client hello, Server hello*

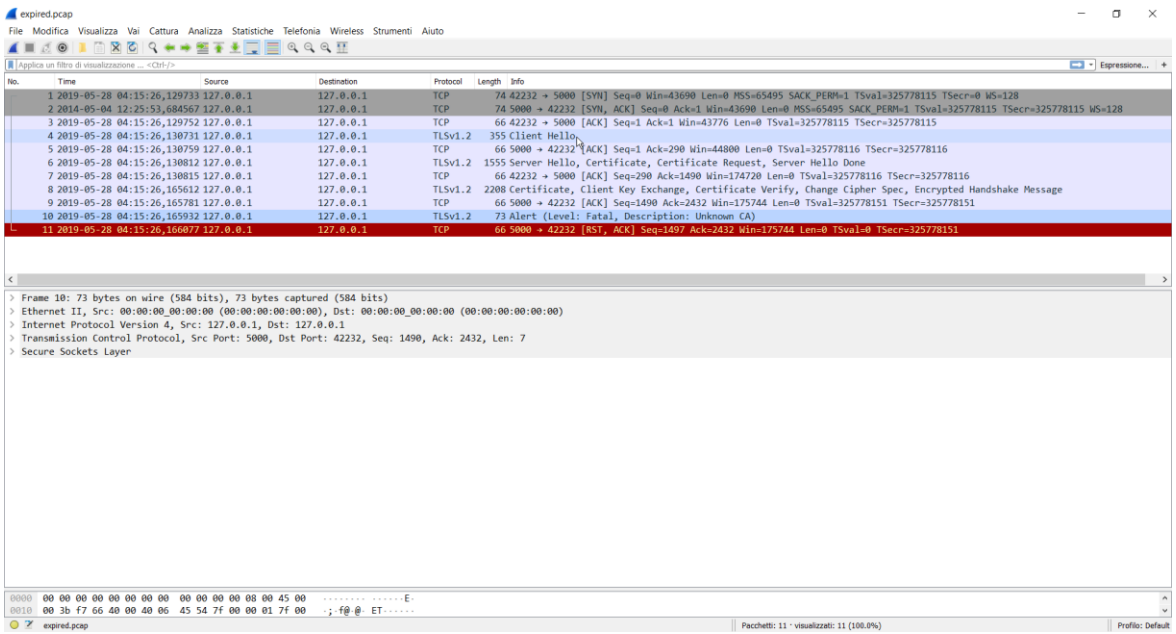

Figure 9.34-2: *Alert Unknown CA*

9.35 Wireshark capture screenshot for test case ID: WINNF.FT.C.SCS.5

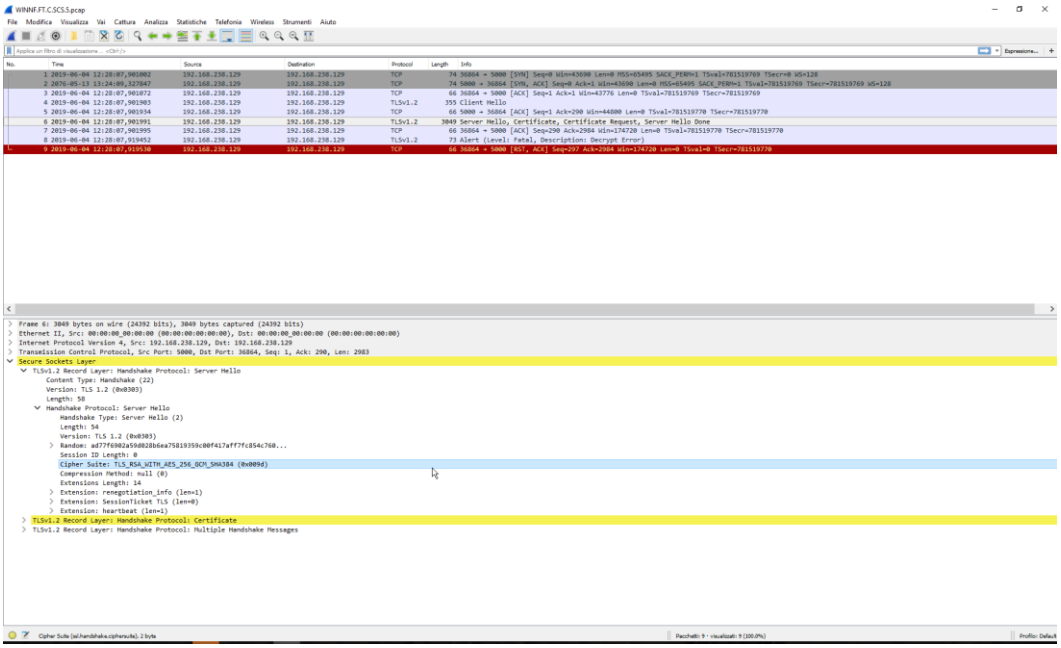

Figure 9.35-1: *Client hello, Server hello, Alert Decrypt Error*

END OF REPORT