



LocoMate User's Guide

Version 1.26

Apr 18, 2013

This document is for informational purposes only and the content is subject to change without notice. Arada MAKES NO WARRANTIES, EXPRESS OR IMPLIED, IN THIS document. All product and service names referenced herein are either trade names or service marks of Arada Corporation in the United States and/or other countries. The other names of actual companies and products mentioned herein may be the trademarks of their respective owners. Copyright 2006,2007 Arada Corporation. All rights reserved. The information contained in this document is subject to change without notice. Reproduction, adaptation or translation without prior permission is prohibited except as allowed under the copyright law.

Table of Contents

Preface	v
Audience/Related Information/contact Us	v
Conventions /Abbreviations & Definations.....	vi
1. Overview of LocoMate	1-1
1.1 Introducing LocoMate	1-1
1.1.1 Contents of the Package.....	1-2
1.1.2 Software Components of LocoMate	1-2
1.2 Connecting LocoMate to the Host Through Telnet	1-3
1.3 Checking the default transmitted data on LocoMate.....	1-3
1.4 Default Configuration	1-3
1.4.1 On Board Unit (OBU) – (or HIA per USDOT)	1-3
1.4.2 Aftermarket Safety Device (ASD).....	1-4
1.4.3 Road Side Unit (RSU)	1-4
1.5 Wireless MAC Randomization.....	1-5
1.6 Radio Monitoring	1-5
1.7 Transferring files from USB storage device	1-5
2. Model Deployment	2-7
2.1 Operating State handling.....	2-7
2.1.1 Initial to Operate.....	2-7
2.1.2 Operate to Quiet	2-7
2.1.3 Operate to Halt	2-7
2.1.4 Operate to No Power	2-7
2.1.5 Quiet to No Power	2-7
2.1.6 Quiet to Operate	2-7
2.1.7 No Power to Operate	2-7
2.1.8 Halt to Operate	2-7
2.1.9 Halt to No Power	2-7
2.2 ModelDeployment Configuration file.....	2-7
2.3 Using time limited one day certificates.....	2-7
2.4 Local Certificate Management (LCM).....	2-8
2.5 Transmit Packet Logs (11p)	2-9
2.6 Event Logs	2-9
2.7 Safety Alert Configuration	2-9
2.8 Visual Indications (LEDs).....	2-9

2.8.1	Power led:	2-9
2.8.2	WLAN led:	2-10
2.8.3	WPS led:	2-10
3.	SAE & Security Applications.....	3-11
3.1	Sending and Receiving SAE J2735 messages with LocoMate	3-11
3.2	Configure PSID	3-13
3.3	Configure User Priority.....	3-14
3.4	Configure Radio Service Channel	3-14
3.5	Configure WSMP Packet Channel.....	3-14
3.6	Configure WSMP Packet Data Rate	3-15
3.7	Configure WSMP Packet Transmit Power	3-15
3.8	Configure Message Interval	3-15
3.9	Configure BSM Part II Message Interval.....	3-15
3.10	Configure BSM Part II PathHistory Type	3-16
3.11	Configure BSM Part II Vehicle Type	3-16
3.12	Configure BSM Vehicle Width	3-16
3.13	Configure BSM Vehicle Length.....	3-16
3.14	Configure application type (User/Provider)	3-16
3.15	Configure Message type (BSM/PVD/RSA/ICA).....	3-17
3.16	Configure Sending of Safety Supplement messages.....	3-17
3.17	Configure WAAS Status	3-17
3.18	Configure Active message Transmission (Store-and-forward).....	3-17
3.19	Configure Immediate forward (forward of packets received over UDP to DSRC radio).....	3-18
3.20	Configure immediate forward (forward of packets from DSRC radio to UDP)	3-18
3.21	Configure EDCA Parameters	3-19
3.22	Configure P1609.2 Sign/Encryption of Messages	3-20
3.22.1	Sign Messages.....	3-20
3.22.2	Encrypt Messages.....	3-22
3.23	Registering Provider application	3-23
3.23.1	WSA configuration through file.....	3-23
3.23.2	Configure Un-Signed WSA	3-24
3.23.3	Configure Sign WSA.....	3-24
3.23.4	Repeat rate in WSA header.....	3-24
3.23.5	Certificate management	3-25
4.	System Message Logging.....	4-26
5.	Communication Message Logging(11p Packets).....	5-28
5.1	Application Layer logging	5-28

5.2	Wireless Interface layer logging	5-29
5.2.1	Configure radio transmit logging:	5-29
5.2.2	Configure radio receive logging:.....	5-30
5.2.3	Configure radio log separation by direction:.....	5-30
5.2.4	Configure radio log separation per interface :	5-30
5.2.5	Configure radio log file size limit.....	5-30
5.2.6	Configure radio log time limit.....	5-30
5.3	Ethernet packet logging	5-32
5.3.1	Configure Ethernet logging enable/disable:	5-32
5.3.2	Configure Ethernet logging interface name:	5-32
5.3.3	Configure Ethernet log separation by direction:.....	5-32
5.3.4	Configure Ethernet log file size limit.....	5-32
5.3.5	Configure Ethernet log time limit.....	5-32
5.4	Log file flush control	5-33
5.5	Log files offload	5-34
5.5.1	Using firmware default keys	5-34
5.5.2	Generating new keys	5-34
6.	Bluetooth Applications.....	6-37
7.	Configuring LocoMate.....	7-38
7.1	Command Line Interface (CLI)	7-38
7.2	Show configuration.....	7-38
7.3	Configuring AP name	7-39
7.4	Operational State change.....	7-39
7.5	Setting password.....	7-39
7.6	Wireless MAC Address Randomization	7-40
7.7	Application profile creation.....	7-40
7.8	Switching between default configured applications.....	7-40
7.9	Configuring a new Application to be executed by default	7-41
7.10	LCM Configuration	7-41
7.11	WAAS Configuration	7-42
7.12	Configure heartbeat message transmission.....	7-42
7.13	Startup script commands.....	7-42
7.14	Network configuration	7-42
7.14.1	Configure number of bridges	7-43
7.14.2	Configure interface for the second bridge.....	7-43
7.14.3	Updating IPv4 Address of LocoMate	7-43
7.14.4	Updating IPv6 Addresses of brtrunk	7-43

7.14.5	Updating IPv6 Addresses of brwifi	7-43
7.14.6	Creating ipv4-to-ipv6 tunnel.....	7-44
7.15	Restore factory Default of LocoMate	7-44
7.16	Restore factory Default of LocoMate through udp.....	7-44
7.17	Rebooting LocoMate from CLI.....	7-44
7.18	Backup Configuration	7-44
7.19	Restore Configuration	7-45
8.	Upgrading LocoMate software.....	8-46
8.1	Firmware Contents	8-46
8.2	Checking the release name (version)	8-46
8.3	Upgrading the firmware using SCP	8-46
8.4	Upgrading the firmware using FTP	8-46
8.5	Upgrading the firmware using file on locomate	8-47
8.6	Configuring TFTP server	8-47
8.7	Upgrading the firmware using TFTP	8-47
8.8	Updating the Security Certificate files through TFTP	8-47
8.9	Updating Active message configuration files	8-48
8.10	Updating Heartbeat message configuration file through TFTP	8-48
8.11	Updating logoffload configuration file through TFTP	8-48
8.12	LocoMate TFTP Commands.....	8-48
8.13	LocoMate FTP Commands	8-49
8.14	Upgrading Firmware through Redboot	8-49
8.14.1	Configuring the ip address at redboot.....	8-50
8.14.2	Update the Flash	8-50
9.	Using WaveDemo	9-51
9.1.1	Setting up a User for Receiving Data.....	9-52
9.1.2	Setting Up a Provider for Transmitting Data.....	9-55
9.2	Running WaveDemo from a Remote Host	9-56
A.1	Wave Radio Stack	A.1-1
A.2	Overview of the Wave Radio Stack	A.2-1
A.3	Components of the Stack.....	A.3-1
B.	IEEE 1609.1: WAVE Resource Manager.....	B-2
B.1	IEEE 1609.2: 5.9 GHz Intelligent Transportation System (ITS) Radio Service Security	B.1-2
B.2	IEEE 1609.3: WAVE Networking Services	B.2-2
B.3	IEEE 1609.4: WAVE Multi-Channel Operation.....	B.3-2
B.4	IEEE 802.11p: Wireless LAN Medium Access Control (MAC) and physical layer (PHY) specifications: Wireless Access in Vehicular Environments (WAVE).....	B.4-2
B.5	WAVE management entity (WME).....	B.5-2

B.6	Wave Short Message Protocol (WSMP).....	B.6-3
C.	Command Reference	C-1
C.1	Using iwconfig	C.1-1
C.2	Using wlanconfig.....	C.2-2
C.3	Private (non-standard) Driver Commands	C.3-3
	List of wavemodeiwpriv Commands.....	C.3-3
D.	Sample configuration files.....	D-5
D.1	Active message configuration file (store-and-forward).....	D.1-5
	TIM data file to be present in Locomate.....	D.1-5
D.2	Immediate message forward configuration file	D.2-7
	MAP data file to be sent over UDP	D.2-7
	SPaT data file to be sent over UDP	D.2-7
	Sample script to send the configuration files	D.2-8
D.3	ModelDeploymentRemovable configuration file.....	D.3-9
D.4	GPS configuration file	D.4-10
D.5	Locomate Safety Alert configuration file.....	D.5-11
	D.5-11
E.	LCM Files	E-12
E.1	Sample ROOT cert file contents.....	E.1-12
E.2	Sample LCM Configuration File	E.2-13
E.3	Sample LCM log file	E.3-14
F.	Configuration for point-to-point setup	F-15

Preface

LocoMate User's Guide helps you to configure and use LocoMate. The document provides an overview of LocoMate and outlines the steps to configure GPS, upgrade firmware, attach external antennas, and replace mini-PCI cards.

Audience

This manual is intended for users who install and configure LocoMate. It is assumed that the user knows the basics of networking, basic linux operations, ip addressing, and wlan fundamentals.

Related Information

For API-related information, refer to *LocoMate Programmer's Guide*.

Contact Us

If you purchased a service program from Arada, you can get help at any time by calling Arada:

+91-80-22107174

Arada would like to know if you found the document useful. If you have any feedback or comments on this document, send us email at:

support@aradasystems.com

Please mention the software, version of the software, and the title of the document in your message.

You can also send us your comments by mail at:

Arada Systems, Inc.
4633 Old Ironsides Drive, Suite
415, Santa Clara, CA 95054,
United States of America (USA)

Conventions

The following conventions are used in the document to help you identify special terms.

Convention	Usage	Example
Bold	The following screen elements: Button List Drop-down menu	Click OK .
<i>Italic</i>	Book titles and emphasis	refer to <i>Concepts Guide</i> for more information.
monospace	Code samples and commands	To check the ip enter the following command: arada# show ip
< >	Mandatory parameters	arada# show card <unit number>
[]	Optional parameters	arada# configure ip address <ip-address> [subnet mask]
	Mutually exclusive choices in a command or code	arada#reboot 1 2 3

Abbreviations and Definitions

The following abbreviations are used in the document

Acronym	Definition
CLI	Command Line Interface
GPS	Global Positioning System
MAC	Medium Access Control
Mb	Megabits
NMEA	National Marine Electronics Association
OBU	On Board Unit
PSID	Provider Service identifier
RSSI	Receive Signal Strength Indication
UDP	User Data Protocol
WAVE	Wireless Access in Vehicular Environments

Acronym	Definition
WBSS	WAVE Base Subscriber Station
WME	WAVE Management Entity
WSMP	Wave Short Message Protocol

1. Overview of LocoMate

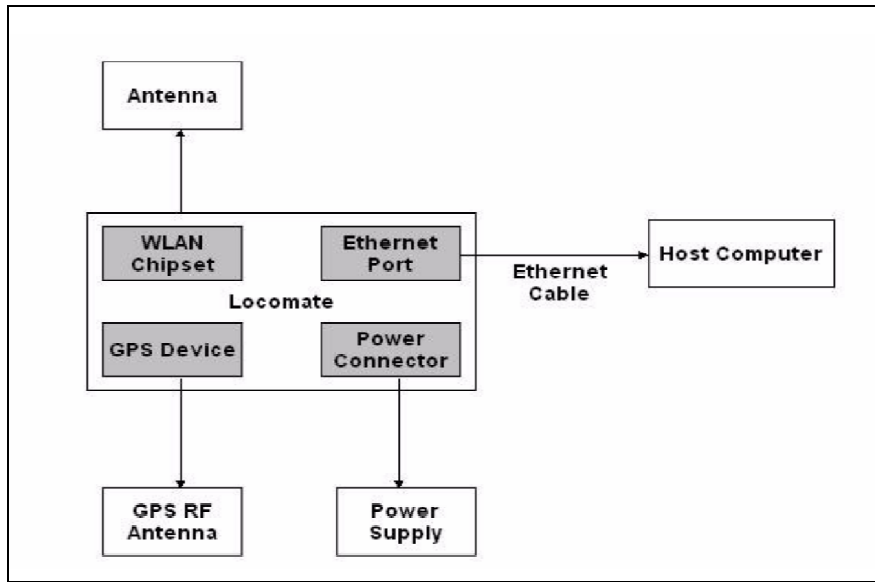
1.1 Introducing LocoMate

LocoMate provides wireless connectivity in automobile environment with high rate and low latency communication between vehicles or between vehicle and road-side unit. It helps provide safety and data services to the vehicle users. It has integrated GPS device, which can be used as a navigation device for the vehicle. By default, the device transmits its position data in continuous service channel (5.86GHz), encoded as per BasicSafetyMessage format. [Figure 1.1](#) shows LocoMate.

Figure 1.1 LocoMate



Figure 1.2 Block diagram of LocoMate with Host computer connection



1.1.1 Contents of the Package

LocoMate package contains:

- MIPS processor running at 680MHz
- Flash – 16MB
- SDRAM – 64MB
- Gigabit Ethernet Interface
- Atheros AR5414 based WLAN Mini PCI
- Output power @ 23dBm
- Integrated GPS device, with external RF antenna

1.1.2 Software Components of LocoMate

LocoMate ships with the following software components:

- Default configured ASN applications: (Please refer to Section 1.4 for more details on RSU/OBU)
 - Application 1: Enabled to execute by default
 - Registers a user application with
 - PSID = 32 (Hex value 0x20)
 - Unconditional Continuous mode of operation
 - Unconditional service channel 172
 - User Priority 2
 - Transmits BasicSafetyMessage encoded packet with GPS information
 - Packet is transmitted in Channel 172
 - Application 2: Configured but disabled to execute by default
 - Registers a user application with
 - PSID = 32 (Hex Value 0x20)
 - Unconditional alternative mode of operation
 - Unconditional service channel 172
 - Transmits BasicSafetyMessage encoded packet with GPS information
 - Packet is transmitted in Channel 178
- Local applications (applications residing in LocoMate and mips executables)
 - localtx
 - Registers a provider with psid=5, service channel (SCH)=172.
 - Creates wsmppackets, and requests the driver to transmit.
 - localrx
 - Registers a user with psid=5.
 - Polls for rx packet

- Bluetooth applications:
 - bluetoothtx
 - bluetoothrx
- Remote applications: (applications residing on remote host with LocoMate communication over ethernet)
 - remotetx
 - Registers a provider with psid=5, service channel (SCH)=172.
 - Creates wsmpp packets, and requests the driver to transmit.
 - Remoterx
 - Registers a user with psid=5
 - Polls for rx packet
- Shared library with APIs to access the DSRC radio and WAVE stack
- toolchain for cross compilation of custom developed applications

1.2 Connecting LocoMate to the Host Through Telnet

To connect LocoMate to a computer:

1. Connect the GPS RF antenna.
2. Connect an Ethernet cable to a host.
3. Configure the host Ethernet IP address to: 192.168.0.100
4. Power on LocoMate.
5. Telnet to the default IP address of LocoMate, which is:192.168.0.40.
Login with the username as 'root', and password as 'password'.

1.3 Checking the default transmitted data on LocoMate

LocoMate contains default application *getwbsstxrrencdec* running. Therefore to configure one of the LocoMate for receive, kill the application with the following procedure:

```
telnet 192.168.0.40 (its default IP address, please use the locomate IP if changed)
```

```
username: root
password: password
```

```
$prompt> ps
```

Note the process id (pid) of the application '*getwbsstxrrencdec*' and execute the following

```
$prompt> kill <pid value>
```

Start the receive application *getwbsstxrrencdec* with PSID=32 (0x20), and service channel 172, with the following command:

```
$prompt> getwbsstxrrencdec -o NOTX -u 2 -x 1 -s 172 -y 32
```

1.4 Default Configuration

- Default IPv4 address:
 - 192.168.0.40
- Default IPv6 address:
 - fe80::200:ff:fe00:0/64

1.4.1 On Board Unit (OBU) – (or HIA per USDOT)

- Enables transmit of BasicSafetyMessage packets
- The wireless MAC address is randomized upon powerup
- Registers a user application with
 - PSID = 32 (Hex Value 0x20)
 - Unconditional Continuous mode of operation

- Unconditional channel 172
- Transmits BasicSafetyMessage encoded packet with GPS information
- Packet is transmitted in Channel 172
- Packet is signed
- Transmit packets are logged onto USB storage device in *pcap* format, with application log enabled. The logging file is created for every 10minutes, with file name format as:
`{year}{month}{date}_{hour}{minute}_{psid(hex value)}_{deviceid(hex value)}_{apname_last_four_characters}_{sequence no}.pcap`
 - For example, with current UTC time as: Aug 03, 2011 19hours 37mins, psid:(32)₁₀, deviceid:0x1234, apname:Automotive
 - The filename is: 20110803_1930_20_1234_TIVE_0001.pcap //first sequence file

1.4.2 Aftermarket Safety Device (ASD)

- Enables transmit and reception of BasicSafetyMessage packets
- Enables reception of Traveler Information Message packets
- Enables processing of received packets for Safety Alerts with option -Y 9999
- Enable ip service user application to join a provider (RSU) with IP service
- The wireless MAC address is randomized upon powerup
- Registers a user application with
 - PSID = 32 (Hex Value 0x20)
 - Unconditional Continuous mode of operation
 - Unconditional channel 172
- Transmits BasicSafetyMessage encoded packet with GPS information
- Packet is transmitted in Channel 172
- Packet is signed
- LCM feature is enabled by default, and can be disabled through CLI configuration as given in Section 2.4 and Section 7.10
- Transmit packets are logged onto USB storage device in *pcap* format, with wireless interface layer logging enabled.. The logging file is with file name format as:
`{year}{month}{date}_{hour}{minute}_{psid(hex value)}_{deviceid(hex value)}_{apname_last_four_characters}_{sequence no}.pcap`
 - For example, with current UTC time as: Aug 03, 2011 19hours 37mins, psid:(32)₁₀, deviceid:0x1234, apname:Automotive
 - The filename is: 20110803_1930_20_1234_TIVE_0001.pcap //first sequence file

1.4.3 Road Side Unit (RSU)

- Registers an application with default parameters taken from active message configuration file (/var/AML/activemsg.conf)
 - The default active message conf file has the following settings:
 - PSID = 32771 (Hex Value 0x8003)
 - Alternate channel mode of operation
 - Transmits Traveler Information Message packet with data as provided in MessagePayload
 - Packet is transmitted in Channel 178 at priority=2 for every 0.1sec
 - Packet is signed
 - Message delivery starts at: 11/01/2011, 00:00
 - Message delivery stops at: 11/30/2014, 23:59
- Transmit and receive packets are logged onto USB storage device in *pcap* format, with wireless interface layer logging enabled.. The logging file is with file name format as:
`{year}{month}{date}_{hour}{minute}_{psid(hex value)}_{deviceid(hex value)}_{apname_last_four_characters}_{sequence no}.pcap`
 - For example, with current UTC time as: Aug 03, 2011 19hours 37mins, psid:(32)₁₀, deviceid:0x1234, apname:Automotive
 - The filename is: 20110803_1930_20_1234_TIVE_0001.pcap //first sequence file
- Registers another application to start an ip service with psid=0x23. The wsa configuration details are taken from the configuration file as in 3.23.
- Registers two applications to start accept data over UDP at ports 4587 and 12345. The received data is forwarded over the DSRC radio as per the configuration in ascii format. Sample configuration files are listed in Chapter D.
- Above mentioned two UDP applications are registered on second radio with default channel of 172.

- By default following log options are enabled:
 - #config log radio-transmitlog enable
 - #config log radio-receivelog enable
 - #config log radio-perdirection enable
 - #config log radio-perinterface enable
 - #config log radio-logfilesize 10 //10MegaBytes(MB)
 - #config log time-based-syslog enable
 - #config log syslog-rotate-time 2330 //23:30
 - #config log ethernetlog disable
 - #config log ethernetlog-perdirection disable
 - #config log ethernet-logfilesize 10 //10MegaBytes(MB)
 - #config log log-file-flush enable

For changing of default application or switching of default application to Application 2, please refer to Section 7.8.

- We can evaluate the application (*getwbsstxrrencdec*) with different configurable parameters, as discussed in Chapter 2.
- The operational state can be configured to ‘*Halt*’ or ‘*Run*’ through the commands and procedure listed in Section 7.3.

Note: It is necessary to kill the default running application before trying to evaluate different options, as we cannot execute two applications with same PSID.

1.5 Wireless MAC Randomization

LocoMate randomizes the wireless mac address at every power up and also randomizes at every certificate change.

1.6 Radio Monitoring

We can check the radio channel status with the following command:

```
$prompt> iwconfig
```

It would display the current active channel, and the current rate, power used in the service channel communication. The channel information will be either control channel value (178) or service channel value depending on the time instance of query. The rate will be the rate used for transmit of IP data packets. The power is the power in dBm used for transmit of data packets. The output power will be much higher than displayed in ‘*iwconfig*’ output due to our hardware manufacturing calibration values.

1.7 Transferring files from USB storage device

To transfer files from the USB storage device to a host system over gigabit ethernet interface, the following steps have to be executed:

1. Setup an *ftp* server (or) a *tftp* server on the host
 - a. It is advisable to chose a host system with gigabit ethernet support for faster transfer of files
 - b. The device
2. Execute ‘*apphalt*’ in cli as mentioned in Section 7.1.
3. Mount the USB storage device on the board (Locomate) in a telnet session
 - a. *\$prompt> mount -t /dev/sda1 /tmp/usb*
 - assuming /tmp/usb directory exists, else create a directory with command :
 - *\$prompt> mkdir /tmp/usb*
 - b. *\$prompt> ls /tmp/usb* should display the contents of the storage device
 - c. Transfer the required files using:
 - FTP Protocol: Refer to section for ftp command details
 - TFTP Protocol: Refer to section 8.12 for tftp command details
4. Our statistics for file transfer time over a gigabit ethernet to a Dell laptop running ubuntu are:

File size	Protocol	10/100 Ethernet	Gigabit Ethernet
-----------	----------	-----------------	------------------

100MB	ftp	10 seconds	8seconds
1GByte	ftp	130 seconds	93seconds

5. The USB storage requires to be unmounted after completion of the access, with the following
 - a. *\$prompt> umount /tmp/usb*

2. Model Deployment

This section summarizes the features relevant to Model Deployment configurations devices

2.1 Operating State handling

Locomate supports the following state transistions in the Model Deployment Locomate devices:

2.1.1 Initial to Operate

The Locomate device shall be activated upon power supply

2.1.2 Operate to Quiet

The Locomate device shall move to quiet state within 12seconds, upon ignition off condition through Vehicle Power Interface - Delphi micro HVT connector. The devices takes care of graceful closing of the log files and event log files into the USB.

2.1.3 Operate to Halt

The Locomate device shall enter halt state through command line interface command as detailed in Section 7.4

2.1.4 Operate to No Power

The Locomate device shall shut off every thing upon power loss in operating state.

2.1.5 Quiet to No Power

The Locomate device shall shut off every thing upon power loss in quiet state.

2.1.6 Quiet to Operate

The Locomate device shall be activated upon ignition on condition.

2.1.7 No Power to Operate

The Locomate device shall be activated upon ignition on condition or power supply condition

2.1.8 Halt to Operate

The Locomate device shall be activated upon issuing ‘apprun’ command from the command line interface as detailed in Section 7.4

2.1.9 Halt to No Power

The Locomate device shall be switched off upon power loss

2.2 ModelDeployment Configuration file

Locomate software expects the configuration file ‘*ModelDeploymentRemovable.conf*’ to be present in the USB storage root directory at:

ModelDeploymentConfigurationItems/

We presently support ‘Version 0.4’ of Vehicle Awareness Device configuration format, as listed in Appendix D.3.

2.3 Using time limited one day certificates

As per USDOT safety pilot program it would be necessary to dynamically use short time limited certificates. We support this feature having all the certificates copied to an USB based flash storage device, along with root_ca.cert file. These files are expected to be in the USB storage root directory as:

- **Note: LCM configuration has to be disabled, for use of time limited one day certificates. LCM feature is enabled by default, and can be disabled through CLI configuration as given in Section 7.10, and reboot the unit.**

2.4 Local Certificate Management (LCM)

There exists LCM module in the Locomate software to enable downloading of certificates from SCMS server. The LCM configuration is enabled by default for After Market Safety Devices (ASD), and can be disabled as detailed in Section 7.10. The operation of LCM requires USB to be plugged in before device power on, and LCM status should be enabled.

LCM module creates 'lcm' folder in USB root directory. On Locomate USB device is default mounted at '/tmp/usb', and hence we refer to the USB root directory accordingly in this guide.

LCM operation requires default 'root_ca.cert' file, in order to join existing IP service application announced in WSA, by an RSE. Root certificate should be placed at: '/tmp/usb/lcm/Certificates'

Note: Locomate software has default root certificate issued by SCMS test server, and is copied to the above folder if there does not exist any 'root_ca.cert' file during bootup.

LCM Config file is at: '/tmp/usb/lcm/lcm.conf'

LCM log file is at: '/tmp/usb/lcm/lcm.log'

LCM downloaded Batch files are at: '/tmp/usb/lcm/'

LCM decrypted certificates are placed at: '/tmp/usb/lcm/Certificates'

Note: Any changes to the lcm configuration file has to be in the following sequence:

- **Transition to 'Halt' state, as in Section 2.1.3**
- **mount/change/unmount usb device with following command on locomate login prompt:**
 - \$prompt>mount -t vfat /dev/sda1 /tmp/usb
 - // Change the lcm.conf file in /tmp/usb/lcm/lcm.conf
 - \$prompt>umount /tmp/usb
- **Transition to 'Run' state, as in Section 2.1.8**
- **To restart the bootstrap process, delete all the files in 'lcm' USB folder except for the new 'lcm.conf' file**
- **Lcm.conf has default log enabled for all the communication messages, except for printing the entire CERTSTATUS_CFM data. To enable logging of this data, enable the value of flag as 'Log_CertStatus_Confirm_Data=1'**

Please find sample root certificate file, LCM config file, LCM log file in Appendix E.3

Note: LCM configuration has to be disabled, for use of time limited one day certificates. LCM feature is enabled by default, and can be disabled through CLI configuration as given in Section 7.10, and reboot the unit.

2.5 Transmit Packet Logs (11p)

The transmit packet logs are stored in the directory '*ModelDeploymentPktCaptures*'. The transmit packet logs are stored in *libpcap* format and arranged as discussed in Section 5.

These organized files are expected to be in the USB storage root directory as:

ModelDeploymentPktCaptures

2.6 Event Logs

Locomate stores the records to formatted files on the transition between the system states of Run, Quiet, and Start, upon the insertion or unmounting of the removable media, and upon each attempt, successful or unsuccessful, to load a configuration. The configuration file '*ModelDeploymentRemovable.conf*' is expected to be present in the directory "*ModelDeploymentConfigurationItems*", for generation of the event logs.

These organized files are expected to be in the USB storage root directory as:

ModelDeploymentSystemEventLogs

2.7 Safety Alert Configuration

Locomate has a choice of different Alert indications as per the following:

- Bluetooth – Alert the visual and voice message through an Android device. This requires installation of Arada_Safety_Application.apk on the Android device
- LED – Alert visual indication by blinking all the three LEDs, as detailed in Section 2.8.1.
- Info - Informational message displayed on application console (used only for debug purposes)

These options can be configured through the file at `"/var/locomateoptions.conf"`. Sample configuration file is as given in Section D.5.

Note: Any changes to the locomate configuration file has to be in the following sequence:

- Transition to 'Halt' state, as in Section 2.1.3
- Change the `/var/locomateoptions.conf` file
- Transition to 'Run' state, as in Section 2.1.8

2.8 Visual Indications (LEDs)

Locomate has three green LEDs, indicating the status of the device. The LEDs are located at the front of the unit next to Ethernet ports. The LEDs are marked as 'WLAN', 'WPS', 'Power'.

Each of the LED status has the following meaning:

2.8.1 Power led:

- a. Glowing: Power input indication
- b. Off: No Power
- c. Blinking: This LED will blink during firmware upgrade process

- d. Blinking along with WLAN, WPS LED: Safety alert is indicated due to forward collision warning, emergency electronic brake light, curve speed warning

2.8.2 WLAN led:

- a. Glowing: WLAN radio is up
- b. Off: WLAN radio is not up
- c. Blinking: WLAN radio is transmitting packets

2.8.3 WPS led:

- a. Glowing: 'ModelDeploymentRemovable.conf' is successfully read by the device firmware and is continuing to access
- b. Off: 'ModelDeploymentRemovable.conf' is not accessible by the device firmware
- c. Blinking: WLAN radio is receiving packets

3. SAE & Security Applications

3.1 Sending and Receiving SAE J2735 messages with LocoMate

The SAE J2735 DSRC Message Set Dictionary defines the application level messages that are exchanged; SAE defines these messages in the ASN.1 format. These application level message sets are for safety messages and will be adopted for interoperability between vendors. Our application *getwbsstxrrencdec* is an integrated application supporting different message sets and default PSID = 32 (Hex Value 0x20).

Few examples on the usage of the application are:

1. Alternative Channel Mode

a) BSM –Basic safety Message

- Transmit
getwbsstxrrencdec -s 172 -t BSM -o norx
 - Registers and Starts a provider
 - Starts channel switch between control channel 178 and service channel 172
 - Transmits the BSM which is encoded in DER on service channel 172
- Receive
getwbsstxrrencdec -w User -s 172 -o notx
 - Registers a user with PSID = 32 (Hex value 0x20) (default value)
 - Decodes the received WSM data and displays in *xml* format

b) Other DSRC Messages supported are, as follows, and they can be selected by giving the corresponding three letter word to ‘-t’ option

- PVD- Probe Vehicle Data
- RSA- Road Side Alert
- ICA – Intersection Collision Alert
- SPAT- Signal Phase And Timing
- MAP- Map Data
- TIM-Traveler Information Message

2. Continuous Channel mode

a. BSM –Basic safety Message

- Provider
getwbsstxrrencdec -s 172 -t BSM -o norx -a 0
 - Registers and Starts a provider
 - Starts continuous channel at service channel 172
 - Transmits the BSM which is encoded in DER on service channel 172
- User
getwbsstxrrencdec -w User -s 172 -o notx -u 2 -x 1
 - Registers a user with PSID = 32 (Hex value 0x20) (default value)
 - Decodes the received WSM data and displays in *xml* format

b. Other DSRC Messages supported are, as follows, and they can be selected by giving the corresponding three letter word to ‘-t’ option

- PVD- Probe Vehicle Data
- RSA- Road Side Alert
- ICA – Intersection Collision Alert
- SPAT- Signal Phase And Timing
- MAP- Map Data
- TIM-Traveler Information Message

NOTE: With use of option 'norx', the application matches its BSM blob temporary id with received packets and regenerates the temporary id. Use of option 'norxall' disables receive of all the packets to the application layer and hence does not match temporary id.

Syntax for the application *getwbsstxrxcdec* is as follows:

usage: getwbsstxrxcdec

***** Common Options *****

- m: Mac Address [xx:xx:xx:xx:xx:xx]
- s: Service Channel
- H: CAN Interface [vcan0/can0/can1]
- b: TxPkt Channel
- w: Service Type [Provider/User]
- t: Message Type [BSM/PVD/RSA/ICA/SPAT/MAP/TIM]
- e: Security Type [Plain/Sign/Encrypt]
- D: Certificate Attach Interval in millisec should be in multiple of packet delay
- l: Output log filename, (specify path ending with / for pcap format)
- o: Tx/Rx Options [TXRX/NOTX/NORXALL/NORX/TXRXUDP/NOTXRX]
- X: Logging Options [TXRXLOG/TXLOG/RXLOG/NOLOG]
- g: sign certificate type [certificate/digest_224/digest_256/certificate_chain]
- p: BSM Part II Packet interval (n BSM Part I messages)
- v: Path history number [2 represents BSM-PH-2, 5 represents BSM-PH-5]
- k: Vehicle_Type (value as per DE_VehicleType)
- y: psid value (any decimal value)
- d: packet delay in millisec
- q: User Priority 0/1/2/3/4/5/6/7
- j: txpower in dBm
- M: Model Deployment Device ID
- T: Temporary ID control (1 = random, 0 = fixed upper two bytes)
- S: Safety Supplement (wsmp-s) <0:disable / 1:enable>
- L: Vehicle Length in cm
- W: Vehicle Width in cm
- r: data rate {0.0, 3.0, 4.5, 6.0, 9.0, 12.0, 18.0, 24.0, 27.0, 36.0, 48.0, 54.0}mbps
- n: no argument, and selects no gps device available
- f: Type xml or csv for logging in XML or CSV format. Type pcaphdr for only pcap header logging & pcap

for full packet logging

- F: frameType for TIM Packet 0-unknown(default) 1-advisory 2-roadSignage 3-commercialSignage
- A: Active Message Status
- B: Port Address for RSU receive from UDP Server
- R: Repeat rate for WSA frame (Number of WSA per 5 seconds)
Repeatrate is included in WSA-Header only if enabled from /proc/wsa_repeatrate_enable
- G: Repeat rate for TA frame (Number of TA per 5 seconds)
TA is available only if TA channel [-c option] is given
- I: IP service Enable 1= enable 0 = disable
- O: Timeout for receiving udp data , in seconds
- Y: Value 0 = xml-print; Value[1 - 9998] = Display Received packet Stats interval in seconds; Value 9999 = Enable Safety Alert processing;
- V: Enable Full-Position Vector Transmit <0:disable / 1:enable>

***** Provider Options *****

- z: Service Priority
- a: Service Channel Access [1:Alternating, 0:Continuous]
- c: Specify Channel Number to Transmit TA
- i: TA Channel Interval [1:cch int, 2:sch int]

***** User Options *****

- u: User Request Type [1:auto, 2:unconditional(not wait for WSA from provider), 3:none]
- x: Extended Access <0:alternate /1:continuous>

Default values:

- m: Mac Address [00:00:00:00:00:00]
- s: Service Channel - 172
- H: CAN Interface - vcan0
- b: TxPkt Channel - 172
- w: Service Type - Provider
- u: User Request Type - [1:auto]
- x: Extended Access - 0
- c: TA disabled - 0
- p: BSM Part II Interval - 10 packets
- k: Vehicle Type - 0 (not available)
- v: PathHistory Number - 4 (PathHistory Set 4)
- t: Message Type - BSM
- e: Security Type - Plain
- l: Output log filename, - NULL
- o: Tx/Rx Options - TXRX
- X: Logging Options - NOLOG
- g: Sign Certificate Type - certificate
- y: psid value - 32
- d: packet delay in millisec - 100
- f: Format Type PCAP
- F: Frame Type 0
- r: data rate(mbps) - 3.0
- A: Active Message Status - 0 (Disable)
- C: Config File Name for active message - '/var/activemsg.conf'
- B: Port Address for RSU receive from UDP Server - 0
- R: Repeat rate for WSA frame - 50
- G: Repeat rate for TA frame - 0
- j: txpower(dBm) - 14
- q: User Priority 2
- S: Safety Supplement (wsmp-s) - 0 (disabled)
- E: Certificate change request flag -0)
- L: Vehicle Length(cm) - 0
- W: Vehicle Width(cm) - 0
- D: Certificate Attach Interval in millisec - 500
- M: Model Deployment Device ID = 1
- T: Temporary ID control = 1 (random 4 bytes)
- I: IP service Enable = 0
- O: Timeout for receiving udp data = 10 seconds
- Y: Display Recv Packet Stats - 0 (xml-print)
- V: Enable Full-Position Vector Transmit - 0 (disable)

3.2 Configure PSID

The application can be configured to use a different PSID with the option '-y'. The default value is 32 (hex 0x20) and can be overridden with decimal value input to '-y' option. The PSID length is as per the P1609 standard, and few examples are:

- Transmit with PSID = 0x22
 - getwbsstxrxcndec -s 172 -t BSM -o norx -y 34*
 - Registers and Starts a provider
 - Starts channel switch between control channel 178 and service channel 172
 - Transmits the BSM which is encoded in DER on service channel 172

- Transmit with PSID = 0x8004
`getwbsstxrxcncdec -s 172 -t BSM -o norx -y 32772`
 - Registers and Starts a provider
 - Starts channel switch between control channel 178 and service channel 172
 - Transmits the BSM which is encoded in DER on service channel 172
- The same option can be used with ‘user’ option as well
- To retain the values across reboot, please update the default application arguments as discussed in Section 7.

3.3 Configure User Priority

The application can be configured to use a different user priority for the data being transmitted with the option ‘-q’. The default value is 2 and can be overridden with decimal value input to ‘-q’ option, in the range of 0 to 7. Few examples are:

- Transmit with user priority = 6
`getwbsstxrxcncdec -s 172 -t BSM -o norx -y 32772 -q 6`
 - Registers and Starts a provider
 - Starts channel switch between control channel 178 and service channel 172
 - Transmits the BSM which is encoded in DER on service channel 172
 - Configure user priority to ‘6’.
- The same option can be used with ‘user’ option as well
- To retain the values across reboot, please update the default application arguments as discussed in Section 7.

3.4 Configure Radio Service Channel

The application can be configured to use a different service channel with the option ‘-s’. The default value is 172 and can be overridden with decimal value input to ‘-s’ option, in the range of

- Supported 10MHz channels: 172, 174, 176, 178, 180, 182, 184
- Supported 20MHz channels: 173, 175, 177, 179, 181, 183.

Few examples are:

- Operate in Service channel 180
`getwbsstxrxcncdec -s 180 -t BSM -o norx -y 34`
 - Registers and Starts a provider
 - Starts channel switch between control channel 178 and service channel 180
 - Transmits the BSM which is encoded in DER on service channel 180
- The same option can be used with ‘user’ option as well
- To retain the values across reboot, please update the default application arguments as discussed in Section 7.

3.5 Configure WSMP Packet Channel

The application can be configured to transmit WSMP messages in either service channel or control channel with the option ‘-b’. The default value is 255 and can be overridden with decimal value input to ‘-b’ option. The valid channel values are as mentioned in the Section 3.4, and with a value of 255 the packet is sent in the current active service channel. Few examples are:

- WSMP data in Control channel 178
`getwbsstxrxcncdec -b 178 -s 180 -t BSM -o norx -y 34`
 - Registers and Starts a provider
 - Starts channel switch between control channel 178 and service channel 180
 - Transmits the BSM which is encoded in DER on control channel 178
- WSMP data in Service channel 180
`getwbsstxrxcncdec -b 180 -s 180 -t BSM -o norx -y 34`
 - Registers and Starts a provider
 - Starts channel switch between control channel 178 and service channel 180
 - Transmits the BSM which is encoded in DER on service channel 180
- The same option can be used with ‘user’ option as well

- To retain the values across reboot, please update the default application arguments as discussed in Section 7.

3.6 Configure WSMP Packet Data Rate

The application can be configured to transmit WSMP messages at a particular data rate as specified by the option ‘-r’. The value can be input in the following format as per the channel in use.

- Possible Data rate options while operating in 10MHz
 - 3.0, 4.5, 6.0, 9.0, 12.0, 18.0, 24.0, 27.0
- Possible Data rate options while operating in 20MHz
 - 6.0, 9.0, 12.0, 18.0, 24.0, 36.0, 48.0 54.0

Few examples of its usage are:

- To transmit WSMP data in Control channel 178 at 3Mbps
 - getwbsstxrxcndec -b 178 -s 180 -t BSM -o norx -y 34 -r 3.0*
 - Transmits the BSM which is encoded in DER on control channel 178 at 3Mbps
- To transmit WSMP data in channel 181 at 54Mbps
 - getwbsstxrxcndec -b 181 -s 181 -t BSM -o norx -y 34 -r 54.0*
 - Transmits the BSM which is encoded in DER on service channel 181 at 54Mbps
- The same option can be used with ‘user’ option as well
- To retain the values across reboot, please update the default application arguments as discussed in Section 7.

3.7 Configure WSMP Packet Transmit Power

The application can be configured to transmit WSMP messages at tx power as specified by the option ‘-j’. The value is in dBm, and also the actual output power is much higher than the specified due to our wireless card design and calibration.

Few examples of its usage are:

- To transmit WSMP data in Control channel 178 at 14dBm
 - getwbsstxrxcndec -b 178 -s 180 -t BSM -o norx -y 34 -r 3.0 -j 14*
 - Transmits the BSM which is encoded in DER on control channel 178 at 3Mbps
- The same option can be used with ‘user’ option as well
- To retain the values across reboot, please update the default application arguments as discussed in Section 7.

3.8 Configure Message Interval

The application can be configured to transmit WSMP messages at an interval as specified by the option ‘-d’. The value is specified in milliseconds and has a default value of 100milliseconds. Few examples of its usage are:

- To transmit WSMP data in Control channel 178 at 3Mbps for every 50msecs
 - getwbsstxrxcndec -b 178 -s 180 -t BSM -o norx -y 34 -r 3.0 -d 50*
 - Transmits the BSM which is encoded in DER on control channel 178 at 3Mbps for every 50msecs
- The same option can be used with ‘user’ option as well
- To retain the values across reboot, please update the default application arguments as discussed in Section 7.

3.9 Configure BSM Part II Message Interval

The application can be configured to transmit BSM Part II WSMP messages at an interval as specified by the option ‘-p’. The value is specified in ‘n’ number of BSM Part I messages and is default to 10. Few examples of its usage are:

- To transmit BSM Part II WSMP data in Control channel 178 at 3Mbps for every 20th BSM Part I message
 - getwbsstxrxcndec -b 178 -s 180 -t BSM -o norx -y 34 -r 3.0 -d 50 -p 20*
 - Transmits the BSM Part II messages, encoded In DER on control channel 178 at 3Mbps for every 20th BSM Part I Message
- The same option can be used with ‘user’ option as well

- To retain the values across reboot, please update the default application arguments as discussed in Section 7.

3.10 Configure BSM Part II PathHistory Type

The application can be configured to transmit BSM Part II WSMP messages with required pathhistory type as specified by the option '-v'. The value is the PathHistorySet number as per the DF_PathHistory field in SAE J2735 document, with default value to 4, for set 4. Few examples of its usage are:

- To transmit BSM Part II WSMP data with PathHistory Set 2, in Control channel 178 at 3Mbps for every 20th BSM Part I message

```
getwbsstxrxcdec -b 178 -s 180 -t BSM -o norx -y 34 -r 3.0 -d 50-p 20 -v 2
```

 - Transmits the BSM Part II messages, encoded In DER on control channel 178 at 3Mbps for every 20th BSM Part I message
- NOTE: The calculations may have to be reviewed in more experiments
- The same option can be used with 'user' option as well
- To retain the values across reboot, please update the default application arguments as discussed in Section 7.

3.11 Configure BSM Part II Vehicle Type

The application can be configured to transmit BSM Part II WSMP messages at an interval as specified by the option '-k'. The value is the DE_VehicleType number as per the DE_VehicleType field in SAE J2735 document, with default value to 0, for unavailable type. Few examples of its usage are:

- To transmit BSM Part II WSMP data with Buses vehicle type, in Control channel 178 at 3Mbps for every 20th BSM Part I message

```
getwbsstxrxcdec -b 178 -s 180 -t BSM -o norx -y 34 -r 3.0 -d 50-p 20 -k 6
```

 - Transmits the BSM Part II messages, encoded In DER on control channel 178 at 3Mbps for every 20th BSM Part I message
- The same option can be used with 'user' option as well
- To retain the values across reboot, please update the default application arguments as discussed in Section 7.

3.12 Configure BSM Vehicle Width

The application can be configured to encode transmit Vehicle width as specified by the option '-W'. The value is the DE_VehicleWidth number as per the DE_VehicleWidth field in SAE J2735 document, with default value to 0, for unavailable width. Example of its usage are:

- To transmit Vehicle width as 30cms, the syntax is as follows, where other options specify channel, message type, psid, rate, packet interval

```
getwbsstxrxcdec -b 178 -s 180 -t BSM -o norx -y 34 -r 3.0 -d 50-p 20 -W 30
```

3.13 Configure BSM Vehicle Length

The application can be configured to encode transmit Vehicle length as specified by the option '-L'. The value is the DE_VehicleLength number as per the DE_VehicleLength field in SAE J2735 document, with default value to 0, for unavailable width. Example of its usage are:

- To transmit Vehicle length as 100cms, the syntax is as follows, where other options specify channel, message type, psid, rate, packet interval

```
getwbsstxrxcdec -b 178 -s 180 -t BSM -o norx -y 34 -r 3.0 -d 50-p 20 -L 100
```

3.14 Configure application type (User/Provider)

The application can be configured to either:

- Start a provider
- Start a user to join a provider
- Start a user with unconditional channel and channel access mode

The application type is specified using the option '-w' with default value as 'Provider', and the possible values are: 'user' and 'provider'.

Few examples are:

- Provider application

```
getwbsstxrxcdec -s 180 -t BSM -o norx -y 34
```

 - Registers and Starts a provider
 - Starts channel switch between control channel 178 and service channel 180
 - Transmits the BSM which is encoded in DER on service channel 180
- The same option can be used with 'user' option as well
- To retain the values across reboot, please update the default application arguments as discussed in Section 7.

3.15 Configure Message type (BSM/PVD/RSA/ICA)

The application can be configured to message data in different message types with the option '-t'. The default value is BSM and can be overridden with value input to '-t' option, with the following supported Messages types :

BSM – Basic Safety Message,
RSA – Road Side Alert
PVD – Probe Vehicle Data
ICA – Intersection Collision Alert

Few examples are:

- Transmit Probe Vehicle Data

```
getwbsstxrxcdec -s 180 -t PVD -o norx -y 34
```

 - Registers and Starts a provider
 - Starts channel switch between control channel 178 and service channel 180
 - Transmits the PVD which is encoded in DER on service channel 180
- The same option can be used with 'user' option as well
- To retain the values across reboot, please update the default application arguments as discussed in Section 7.

3.16 Configure Sending of Safety Supplement messages

Locomate supports sending of Safety Supplement message with the option '-S' and value 1 to enable sending of WSMP-S control field and 0 – to disable sending of WSMP-S byte. The WSMP-S control byte carries the sender state information and is presently supporting to indicate the sender state (0 or 1) which is based on channel access.

3.17 Configure WAAS Status

Locomate supports configuring of WAAS enable/disable to get more accurate gps data using WAAS corrections. For detail configuration steps please refer section 7.11.

3.18 Configure Active message Transmission (Store-and-forward)

The application can be configured to Transmit payload on the basis of active message configuration file.

To configure use option '-A' as 1. The config files are taken from the directory '/var/AML/'

Example:-

```
getwbsstxrxcdec -o NORXALL -w Provider -A 1
```

- Registers and Starts a Provider.
- Look for configuration file and parse the data from it.
- Other Transmission options like '-y','a','-b','-d','-e','q', along with start time and stop time for transmission will be overridden for transmission from config file.
- DER encoded payload will be transmitted between start and stop UTC time.
- All the files in the directory '/var/AML' are expected to be valid active message configuration files
- The application loads all the files in '/var/AML' directory and transmits data as per the configuration

Note:- To load an active message configuration file on Locomate refer section 8.9. A sample configuration file is present at /var/AML and is also listed in section 0.

3.19 Configure Immediate forward (forward of packets received over UDP to DSRC radio)

In immediate forward feature, the data received over the UDP is forwarded over the DSRC radio as per the configuration in ascii format. The application can be configured to accept messages over UDP on a specified port, and are processed to transmit immediately or as per the transmit interval requirements. To configure use option '-o' as TXRXUDP and '-B' providing the port number.

Example:-

```
getwsbstxrxcnec -o TXRXUDP -w User -u 2 -B <port number>
```

- Registers and Starts a user in unconditional mode
- Waits for configuration data over the UDP port
- Other Transmission options like '-y','a','-b','-d','-e','q', along with start time and stop time for transmission will be overridden for transmission from config file.
- -O option configures the timeout (in seconds) for packets to receive over UDP. The default timeout value is 10 seconds

WSMP encoded payload will be transmitted between start and stop UTC time as per the transmit interval in the configuration data. Sample configuration files are listed in sections 0 and 0. Also script to feed the configuration data to Locomate is listed in Section 0.

Our device requires an unique application per psid, and so to enable transmission of map and spat, it is required to configure multiple application profiles. As an example, the below picture shows configuration with four applications (1) active message (2) Immediate fwd application on port 4587 (3) Registering an IP service application with psid 35 (0x23) (4) Immediate fwd application on port 12345

```

arada@localhost:~$ cat /etc/network/interfaces
syslog information
    server-ip      0.0.0.0
    server-port    514
    syslog-status  disable
    transmitlog    enable
    receive-log    disable

ip
    address 192.168.0.40
    netmask 255.255.255.0
    default-gateway 0.0.0.0
    dns-server-primary 0.0.0.0
    dns-server-secondary 0.0.0.0
    dhcp-client disable
    traffic-control-server-connection-check enable
    no-of-bridges 2
    brtrunk-ipv6Status static
    brtrunk-ipv6-address 2001::470::e0fb::1111:::6666/64
    brtrunk-ipv6-gateway 2001::470::e0fb::1111:::aaaa
    brtrunk-ipv6-network-prefix 2000:::/3
    interface-attached-to-second-bridge wifilvap0
    brwifi-ipv6Status static
    brwifi-ipv6-address 2001::470::e0fb::6666:::1/64
    brwifi-ipv6-gateway 0
    brwifi-ipv6-network-prefix 0

snmp
    status disable
    trap-server-ip 0.0.0.0
    trap-server-community trap
    read-community public
    read-write-community private
    trap-port 162

applications
    application    status    Name                Argument
    application1   enable   /usr/local/bin/getwsbstxrxcnec  -o\ NORXALL\ -w\ User\ -u\ 2\ -x\ 0\ -s\ 176\ -A\ 1
    application2   enable   /usr/local/bin/getwsbstxrxcnec  -o\ TXRXUDP\ -w\ User\ -u\ 2\ -B\ 4587\ -y\ 49136
    application3   enable   /usr/local/bin/getwsbstxrxcnec  -o\ NOTTXRX\ -y\ 35\ -a\ 1\ -s\ 176
    application4   enable   /usr/local/bin/getwsbstxrxcnec  -o\ TXRXUDP\ -w\ User\ -u\ 2\ -B\ 12345\ -y\ 49120
  
```

3.20 Configure immediate forward (forward of packets from DSRC radio to UDP)

Locomate software provides an application 'wsmpforward' that forwards specified psid packets to a remote host using UDP.

The syntax of the application is:

```
wsmppforward -i <ipv6-of-remote-host> -p <port> -u <user-req-type> -x  
             <extended access> -y <psid> -s <service channel>
```

***** Options *****

```
-h:      This help  
-i:      IPv6 address of remote host (currently supports only IPv6)  
-p:      Port number of remote host side listener  
-y:      PSID of the service of which packets are to be forwarded  
-u:      User Request Type [1:auto, 2:unconditional, 3:none]  
-x:      Extended Access <0:alternate /1:continuous>  
-s:      Service Channel (Mandatory is user request is unconditional)  
DEFAULT Values for some options [ -u 1 ] [ -y 32] [ -x 0] [ -s 172]
```

We also provide a sample application that can receive the forwarded packets over the UDP and display first 8 bytes of data along with packet len. The application is named 'wsmppforwardserver'. Sample application has to be run on remote host to receive and decode packets transmitted by 'wsmppforward'. The application can be found in the x86 source package, and accessible from support site.

The format of the packet forwarded is as in the structure 'wsm_indication' defined in 'wsmppforwardserver.c'.

3.21 Configure EDCA Parameters

LocoMate support configuring of EDCA parameters for both control channel (cch) and service channel (sch). These values are presently not stored in the database and so will be lost with power recycle. The syntax for reading the current EDCA parameters and configuring the EDCA parameters is as follows:

Getting EDCA params

```
iwpriv ath0 get_aifs <ac> <sch>
iwpriv ath0 get_cwmin <ac> <sch/not sch>
iwpriv ath0 get_cwmax <ac> <sch/not sch>
iwpriv ath0 get_txoplimit <ac> <sch/not sch>
iwpriv ath0 get_acm <ac> <sch/not sch>
```

For Example:

```
iwpriv ath0 get_aifs 1 0 //for getting aifs of WME_AC_BK of control-channel
```

Setting EDCA params

```
iwpriv ath0 aifs <ac> <sch/not sch> <value>
iwpriv ath0 cwmin <ac> <sch/not sch> <value>
iwpriv ath0 cwmax <ac> <sch/not sch> <value>
iwpriv ath0 txoplimit <ac> <sch/not sch> <value>
iwpriv ath0 acm <ac> <sch/not sch> <value>
```

ex:-

```
iwpriv ath0 aifs 1 0 5 // sets aifs of a control-channel for ac WME_AC_BK to 5
```

Note: 'ac', 'sch' and 'not sch' refer to the values as below

>>Access Categories List <ac>

```
>>WME_AC_BE    0      /* best effort */
>>WME_AC_BK    1      /* background */
>>WME_AC_VI    2      /* video */
>>WME_AC_VO    3      /* voice */
```

>>sch=1 // (sch) service-channel

>>sch=0 // (not sch) control channel

3.22 Configure P1609.2 Sign/Encryption of Messages

LocoMate supports two types of security related Messages as per IEEE 1609.2, ie: signed and encrypted. Examples commands for Signed/Encrypted forms of SAE Message types are shown below

3.22.1 Sign Messages

1. Alternative Channel Mode

1. BSM –Basic safety Message

- Transmit

```
getwbsstxrxcdec -s 172 -t BSM -o norx -e sign
```

 - Registers and Starts a provider
 - Starts channel switch between control channel 178 and service channel 172
 - Transmits the BSM which is encoded in DER on service channel 172
- Receive

```
getwbsstxrxcdec -w User -s 172 -o notx
```

 - Registers a user with PSID = 32 (Hex value 0x20) (default value)
 - Decodes the received WSM data and displays in *xml* format

2. Other DSRC Messages supported are, as follows, and they can be selected by giving the corresponding three letter word to 't' option
 - PVD- Probe Vehicle Data

- RSA- Road Side Alert
- ICA – Intersection Collision Alert

1. Continuous Channel mode

1.1 BSM –Basic safety Message

- Provider
 - `getwbsstxrrencdec -s 172 -t BSM -o norx -a 0 -e sign`
 - Registers and Starts a provider
 - Starts continuous channel at service channel 172
 - Transmits the BSM which is encoded in DER on service channel 172
- User
 - `getwbsstxrrencdec -w User -s 172 -o notx -u 2 -x 1`
 - Registers a user with PSID = 32 (Hex value 0x20) (default value)
 - Decodes the received WSM data and displays in *xml* format

1.2 Other DSRC Messages supported are, as follows, and they can be selected by giving the corresponding three letter word to ‘-t’ option

- PVD- Probe Vehicle Data
- RSA- Road Side Alert
- ICA – Intersection Collision Alert

Snapshot of a Signed BSM message receive

```

root@localhost:~
File Edit View Terminal Tabs Help
*****
[root@Arada01206F /root]#
[root@Arada01206F /root]# getwbsstxrrencdec -w User -o NOTX -x 0 -u 2 -t BSM -s 172 -b 172 -e Sign
Connecting to remote IP 127.0.0.1
Connecting ...
Connected.
Inside User process1
Invoking WAVE driver
Registering User 11 app pid = 688
In User end
RX thread
Received WSM Packet Channel = 172, Packet No =#0# Content_type# Signed
Send AsnMsg_Verify request. [0x02]
Receive AsnMsg_Verify response. [0x12]
Signature Algorithm ECDSA-224
<BasicSafetyMessage>
  <msgID><basicSafetyMessage/></msgID>
  <blob>
    00 06 BE 00 00 03 07 BB 04 94 2E 40 E9 A2 00
    00 00 00 00 00 00 70 80 00 00 00 00 00 00 00
    00 00 00 00 00
  </blob>
  <status>
    <fullPos>
      <utcTime>
        <year>11</year>
        <month>8</month>
        <day>4</day>
        <hour>10</hour>
        <minute>22</minute>
        <second>54974</second>
      </utcTime>
      <long>776006050</long>
      <lat>129696916</lat>
      <elevation>24 B9</elevation>
      <heading>28800</heading>
      <speed>00 00</speed>
    </fullPos>
  </status>
</BasicSafetyMessage>
cReceived WSM Packet Channel = 172, Packet No =#1# Content_type# Signed
Send AsnMsg_Verify request. [0x02]
Receive AsnMsg_Verify response. [0x12]
Signature Algorithm ECDSA-224
<BasicSafetyMessage>

```

3.22.1.1 Configuring certificate type

We can configure the Sign Certificate Type to either of the following *[certificate/Digest/Certificate_chain]*, through ‘-g’ option.

3.22.1.2 Configuring certificate attachment interval

We can configure the certificate attachment interval with the option ‘-D’, which specifies certificate attachment for every ‘N’ number of messages. The default value is 20 and a certificate is transmitted every 20 messages, else a digest is attached with the message.

3.22.2 Encrypt Messages

1. Alternative Channel Mode

a) BSM –Basic safety Message

- Transmit
getwbsstxrrencdec -s 172 -t BSM -o norx -e encrypt
 - Registers and Starts a provider
 - Starts channel switch between control channel 178 and service channel 172
 - Transmits the BSM which is encoded in DER on service channel 172
- Receive
getwbsstxrrencdec -w User -s 172 -o notx
 - Registers a user with PSID = 32 (Hex value 0x20) (default value)
 - Decodes the received WSM data and displays in *xml* format

b) Other DSRC Messages supported are, as follows, and they can be selected by giving the corresponding three letter word to ‘-t’ option

- PVD- Probe Vehicle Data
- RSA- Road Side Alert
- ICA – Intersection Collision Alert

2. Continuous Channel mode

a) BSM –Basic safety Message

- Provider
getwbsstxrrencdec -s 172 -t BSM -o norx -a 0 -e encrypt
 - Registers and Starts a provider
 - Starts continuous channel at service channel 172
 - Transmits the BSM which is encoded in DER on service channel 172
- User
getwbsstxrrencdec -w User -s 172 -o notx -u 2 -x 1
 - Registers a user with PSID = 32 (Hex value 0x20) (default value)
 - Decodes the received WSM data and displays in *xml* format

b) Other DSRC Messages supported are, as follows, and they can be selected by giving the corresponding three letter word to ‘-t’ option

- PVD- Probe Vehicle Data
- RSA- Road Side Alert
- ICA – Intersection Collision Alert

Snapshot of a Signed BSM message receive

```

root@localhost:/
File Edit View Terminal Tabs Help
root@localhost:/ root@localhost:/ root@localhost:~
*****
[root@Arada01206F /root]#
[root@Arada01206F /root]# getwsstxrncdec -w User -o NOTX -u 2 -x 0 -t BSM -s 172 -b 172 -e Encrypt
Connecting to remote IP 127.0.0.1
Connecting ...
Connected.
Inside User process1
Invoking WAVE driver
Registering User 11 app pid = 753
In User end
RX thread
Received WSMP Packet Channel = 172, Packet No =#0# Content_type# Encrypted
Encryption Algorithm AES_128_cm
Send AsmMsg_Dec request. [0x04]
Receive AsmMsg_Dec response. [0x14]
<BasicSafetyMessage>
  <msgID><basicSafetyMessage/></msgID>
  <blob1>
    01 2A 2F 00 00 00 03 07 BB 03 58 2E 40 B8 A7 00
    00 00 00 00 00 00 70 80 00 00 00 00 00 00 00
    00 00 00 00 00 00
  </blob1>
  <status>
    <fullPos>
      <utcTime>
        <year>11</year>
        <month>8</month>
        <day>4</day>
        <hour>10</hour>
        <minute>59</minute>
        <second>10799</second>
      </utcTime>
      <long>776005799</long>
      <lat>129696600</lat>
      <elevation>24 10</elevation>
      <heading>28800</heading>
      <speed>00 00</speed>
    </fullPos>
  </status>
</BasicSafetyMessage>
Received WSMP Packet Channel = 172, Packet No =#1# Content_type# Encrypted
Encryption Algorithm AES_128_cm
Send AsmMsg_Dec request. [0x04]

```

3.23 Registering Provider application

3.23.1 WSA configuration through file

WSAs are constructed with several elements filled based on the applications registered to the device. Fields like PSID, channel information are input through the applications registered (Refer 3 and 7.7)

Configurable WSA fields are in the file /var/wsa.conf

following is the sample wsa.conf file. lines beginning with # are comments

fields to be filled by default/zero can be filled as “unavailable” or commented out using # at the start of line

```

-----
# Advertiser Identifier
advertiser_identifier=USDOT

# Provider Service Context
servinfo.psc=SCMS

# IPv6 Address in Service Info
servinfo.ipv6addr=unavailable

# Service Port in Service Info
servinfo.port=unavailable

# IpPrefix in WRA
wra.ipprefix=2001:470:e0fb:6666::

# Prefix Length in WRA
wra.prefixlen=64

```

```
# Default Gateway in WRA
wra.defaultgw=2001:470:e0fb:6666::1
```

```
# Primary DNS in WRA
wra.primaryDNS=2001:470:e0fb:6666::1
```

3.23.2 Configure Un-Signed WSA

By default it is configured to send signed WSA, to enable transmission of unsigned WSAs. It is required to perform the following:

1. Disable the SignWSA as below,
 1. add the following to /var/appstart script at the end and Reboot
- ```
echo 0 0 > /proc/wsmc_sign_wsa
```
- Now RSE sends Unsigned WSA messages on channel 178

### 3.23.3 Configure Sign WSA

We can enable signing of WSA messages and we can also configure the psid and priority of signed WSA messages by executing the following command

```
$prompt> echo 1 <num entries> <psid> <priority> <ssp string "P"> > /proc/wsmc_sign_wsa
```

Example:- If we want to configure Sign WSA messages with different psids Then we have to give following commands.

1. For configuring sign WSA messages with only 1 psid 32 then following command should be used.

When SSP is equal to "P1"

```
<psid>:<priority>:<SSP string> like 32:31:P1
```

```
$prompt> echo 1 1 32 10 2 P 1 > /proc/wsmc_sign_wsa
```

When SSP is equal to NULL

```
<psid>:<priority>: like 32:31:
```

```
$prompt> echo 1 1 32 10 0 > /proc/wsmc_sign_wsa
```

When SSP is equal to "P12"

```
<psid>:<priority>:<SSP string> like 32:31:P12
```

```
$prompt> echo 1 1 32 10 3 P 1 2 > /proc/wsmc_sign_wsa
```

2. The default configured value is following for PSIDs: 32 (0x20), 32771 (0x8003), 49120 (0xbfe0), 49136 (0xbff0), 49121 (0xbfe1)

```
$prompt> echo 1 6 32 31 0 32771 31 0 49120 31 0 49136 31 0 35 31 0 49121 31 0
> /proc/wsmc_sign_wsa
```

Note:- We can configure sign WSA messages with these commands with the same psids and priorities given at the time of security certificate creation. Currently we are using permission where SSP string is NULL.

### 3.23.4 Repeat rate in WSA header

As mentioned in `getwbsstxrxcndec` usage (section 3.1), Repeat rate for WSA frame (Number of WSA per 5 seconds) can be configured with "-R" option.

Repeatrate can be included in WSA-Header by enabling /proc/wsa\_repeatrate\_enable entry as below:

```
$prompt> echo 1 > /proc/wsa_repeatrate_enable //to enable
```

```
$prompt> echo 0 > /proc/wsa_repeatrate_enable //to disable
```

## 3.23.5 Certificate management

### 3.23.5.1 Certificate upload and reload

We support upload of the certificate, private key files through tftp protocol by transferring a certificate tar file to the LocoMate. At the end of update the wireless device mac address is randomized. We can upload new certificates to the LocoMate through the following procedure.

1. Issue ‘*apphalt*’ command at CLI prompt (as mentioned in Section 7.3)
2. Create a ‘*tar*’ file of the necessary certificate files or for example the following certificate files:
  - a. *root\_ca.cert* // Root CA certificate file
  - b. *root\_ca.key* // Root CA private key file
  - c. *msg\_ca.cert* // Message CA certificate file
  - d. *msg\_ca.key* // Message CA private key file
  - e. *msg\_id\_nlo.cert* // Message ID non-localized certificate
  - f. *msg\_id\_nlo.key* // Message ID non-localized private key pair
  - g. *msg\_id\_nlo\_dec.key* // Message ID non-localized decrypt key
  - h. *wsa\_ca.cert* // WSA CA certificate file
  - i. *wsa\_ca.key* // WSA CA private key file
  - j. *wsa\_ca.map* // WSA map file
  - k. *wsa\_sign.cert* // WSA Sign certificate file
  - l. *wsa\_sign.key* // WSA Sign private key file
  - m. *wsa\_sign.map* // WSA sign map file
3. Our security configuration assumes that the
  - a. *msg\_sign* is derived from *msg\_ca* which in turn derived from *root\_ca*
  - b. *wsa\_sign* is derived from *wsa\_ca* which in turn derived from *root\_ca*
4. Update certificate files as per the procedure listed in Section 8.8.
5. Issue ‘*apprun*’ command at CLI prompt (as mentioned in Section 7.3)

### 3.23.5.2 Certificate storage

The certificates can be stored in */var* directory and will be saved across power cycles. The certificates can also be stored in the USB storage disk provided along with LocoMate.

### 3.23.5.3 Time limited certificates

Our security module will check for expired certificates and avoid using them for signing and encryption.

### 3.23.5.4 Certificate deletion upon expiration

This feature is presently not automated and needs to be manually delete the expired files from the storage device

## 4. System Message Logging

LocoMate has a system log daemon running and collects the log messages from different executing components of the system. The logs are collected into the file */var/log/messages*. The log file is default configured for 64K file size limit with log rotation. After every 64kb of logging in */var/log/messages*, the messages generated are moved to a file named '*messages\_utctimestamp*' file with utc timestamp in the form "*yyyymmdd\_hhmm*". The file *messages\_utctimestamp* records events upto the specified utctime in utctimestamp of the message filename.

A new file *messages\_utctimestamp* is created after every 64kb of logging containing latest logs. Locomate device create and use a new active SSL file upon closing the previously active file.

Syslog file maximum size and max no.of file can configure through CLI as below:

1) for syslog-file-size :- This is the size of */var/log/messages* file, after completing this size file will rotate.

*#cli*

*#config log syslog-file-size [filesize(in kb)] // default file size is 64 kb.*

*#exit*

2) for syslog-file-count :- This is the number of old files to be retained in the */var/log/* folder along with messages file.

*#cli*

*#config log syslog-file-count [filecount] // default file count is 2*

*#exit*

e.g. If syslog-file-count is configured to value 3 then there will be 3 *messages\_<UTC Time Stamp>* files along with messages file in */var/log/* folder at any point of time.

In RSE, SSL log file can be generated based on time along with default 64k size criteria. Periodically generated SSL log files in */var/log/messages* are moved to *ssl* folder in USB as */tmp/usb/ssl*. Time Based SSL file generation can be controlled in RSE as below:

3) time-based-syslog //to enable/disable time based syslog rotation

4) syslog-rotate-time //time of day (24 hour format) after which the SSL will be rotated

e.g. To create SSL every day after 23:00

*Prompt#cli*

*#config log time-based-syslog disable*

*#config log syslog-rotate-time 2300*

*#config log time-based-syslog enable*

By default following log options are enabled:

*#config log time-based-syslog enable*

*#config log syslog-rotate-time 2330 //23:30*

*#config log syslog-file-size 64 //64kb*

*#config log syslog-file-count 2*

No reboot is required as above mentioned 4 changes will restart the syslogd runtime.

The log file contains the messages with UTC time at the start of every event message.

Each message in the log file will have UTC time stamp followed by the message:

e.g: *Mar 26 13:25:25 syslog: gpstime: SLT tv=1332768325 - usecs=0 at=1332768325.000000 fix=3*

The log files can be transferred to the host through tftp commands as mentioned in Section 8.3 and 5.5.

Other important log files created are:

*/var/asm.log* – contains the log messages from security module

Kernel Log buffer can be seen with '*dmesg*' command at the telnet console prompt. This buffer is not stored permanently and has to be copied to */var* directory to store the log permanently.

# 5. Communication Message Logging(11p Packets)

## 5.1 Application Layer logging

Arada LocoMate supports logging of the packets at both Transmitter and Receiver side with -X option as follows:  
This is default enabled for hardware with product id as "OBU".

TXRXLOG :- Logs Transmitting & Receiving packets in the same file ( xml and csv format are not supported, only pcap and pcap header format supported).

TXLOG :- Logs the Transmitting packets (only pcap and pcap header format are supported)

RXLOG :- Logs the Receiving packets (xml, csv, pcap are supported)

NOLOG:- Default Option

Logging will be enabled only when suitable option for -X option other than NOLOG is given.  
Logging at transmitting and receiving end can be done as follows.

### 1. Transmitter :

a) Logging with pcap format and log files are stored at location /tmp/ with UTC time-stamp.

b) **USB logging** in pcap format and log files with UTC timestamp are copied to USB every 10 minutes if logfilename specified by -l option is /tmp/usb/ only.

Expected options to enable the logging using the *getwbsstxrxcdec* application can be :

*-f pcap -l /tmp/usb/ -X TXLOG/TXRXLOG*

The files will be logged to a new file for every 10 minutes and the file is created as per the date and time:

- The logging file is with file name format as:
- {year}{month}{date}\_{hour}{minute}\_{psid(hex value)}\_{deviceid(hex value)}\_{apname\_last\_four\_characters}\_{sequence no}.pcap
  - For example, with current UTC time as: Aug 03, 2011 19hours 37mins, psid:(32)<sub>10</sub>; deviceid:0x1234, apname:Automotive

The filename is: 20110803\_1930\_20\_1234\_TIVE\_0001.pcap //first sequence file

Since the default logging is onto the USB device, you should find all the logged packets in the USB device as per the above format.

### 2. Receiver :

a) Logging in xml , csv or pcap format and log files are stored at location /tmp/ with UTC timestamp.

b) **USB logging** in xml or csv or pcap format and log files with UTC timestamp are copied to USB every 10 min if logfilename specified by -l option is /tmp/usb/ only.

Expected options can be :

-o NOTX                      -l /tmp/usb/                      -f xml (use pcap or csv to get desired file format)  
-X RXLOG/TXRXLOG (logs the Rx packets)

Note: for usb logging " -l /tmp/usb/ " option is mandatory

Sample received *xml* file logging is as follows:

```

<XMLLOG>
[BEGIN] <seq=1>
<logtime seconds> 1290529932 </logtime seconds>
<microseconds> 399525 </microseconds>
<src> 00:26:ad:01:17:89 </src>
<type> BSM </type>
<psid> 11 </psid>
<ver> 2 </ver>
<sec> 0 </sec>
<Year>11 </Year>
<Month>8 </Month>
<Day>4 </Day>
<Hour>13 </Hour>
<Min>43 </Min>
<sec>18264 </sec>
<longitude> 776004866 </longitude> <londir> E </londir>
<latitude> 129697033 </latitude> <latdir> N </latdir>
<elevation>9283 <elevation> <speed>0 </speed>
<heading>28800 </heading>
[END]
[BEGIN] <seq=2>
<logtime seconds> 1290529932 </logtime seconds>
<microseconds> 504155 </microseconds>
<src> 00:26:ad:01:17:89 </src>
<type> BSM </type>
<psid> 11 </psid>
<ver> 2 </ver>
<sec> 0 </sec>
<Year>11 </Year>
<Month>8 </Month>
<Day>4 </Day>
<Hour>13 </Hour>
<Min>43 </Min>
<sec>21367 </sec>
<longitude> 776004866 </longitude> <londir> E </londir>
<latitude> 129697050 </latitude> <latdir> N </latdir>
<elevation>9283 <elevation> <speed>0 </speed>
<heading>28800 </heading>
[END]

</XMLLOG>

```

## 5.2 Wireless Interface layer logging

The firmware supports logging all packets at the wireless interface layer, which can be enabled through cli.

This is default enabled for hardware with product id as “ASD” and “RSU”.

To check the status of logging, log on to board using telnet and issue login id and password

Execute “cli” command

Execute “show configuration”

Under syslog information check whether transmitlog is enabled

By default transmitlog(under syslog) is enabled, for RSU

### 5.2.1 Configure radio transmit logging:

Prompt# cli

Prompt# apphalt

Prompt# config log radio-transmitlog enable



Prompt#exit

### 5.2.2 Configure radio receive logging:

```
Prompt# cli
Prompt# apphalt
Prompt# config log radio-receivelog enable
Prompt#exit
```

### 5.2.3 Configure radio log separation by direction:

```
Prompt#cli
Prompt# apphalt
Prompt #config log radio-perdirection enable
Prompt#exit
```

### 5.2.4 Configure radio log separation per interface :

```
Prompt#cli
Prompt# apphalt
Prompt#config log radio-perinterface enable
Prompt#exit
```

### 5.2.5 Configure radio log file size limit

Configures the limit for size of each file created, with default size limit at 10MB

```
Prompt#cli
Prompt# apphalt
Prompt#config log radio-logfilesize <value>
// Where, <value> is in units of MB, with float value indicating KB
// <value> as 10 is taken as (10*1048576) bytes = 10MB
// <value> as 0.4 indicates (0.125 * 1048576) bytes = 128KB

Prompt#exit
```

### 5.2.6 Configure radio log time limit

Configures the time limit for each file created, with default time log disabled.

```
Prompt#cli
Prompt# apphalt
Prompt#config log radio- logfiletime <value>
// Where, <value> is in units of minutes
// <value> as 10 is taken as ten minutes

Prompt#exit
```

Additionally both (per-direction and per-interface) can be enabled simultaneously.

By default following log options are enabled:

```
#config log radio-transmitlog enable
#config log radio-receivelog enable
#config log radio-perdirection enable
#config log radio-perinterface enable
#config log radio-logfilesize 10 //10MegaBytes(MB)
```

#### Log file name table:

|                                                         | <b>TX_LOG<br/>Enable</b>                                                                    | <b>RX_LOG<br/>Enable</b> | <b>BOTH(TX_LOG &amp; RX_LOG)<br/>Enable</b> |
|---------------------------------------------------------|---------------------------------------------------------------------------------------------|--------------------------|---------------------------------------------|
| <b>Direction separation enable</b>                      | Tx                                                                                          | Rx                       | Tx, Rx                                      |
| <b>Interface separation enable</b>                      | wifi0_tx,<br>wifi1_tx                                                                       | wifi0_rx,<br>wifi1_rx    | wifi0_tx_rx, wifi1_tx_rx                    |
| <b>Direction &amp; Interface separation<br/>enable</b>  | wifi0_tx,<br>wifi1_tx                                                                       | wifi0_rx,<br>wifi1_rx    | wifi0_tx, wifi1_tx<br>wifi0_rx, wifi1_rx    |
| <b>Direction &amp; Interface separation<br/>Disable</b> | Generated Log file names contain as below except<br><br>< value as per above table > field. |                          |                                             |

Generated log file name contains: UTCTimeStamp\_ff\_0000\_last 4 characters of apname\_<value as per above table>\_sequenceNumber.pcap

#### Example file names:

- Both Direction & interface separation enabled
  - 20120824\_1650\_ff\_0000\_209A\_wifi0\_rx\_0001.pcap
  - 20120824\_1650\_ff\_0000\_209A\_wifi0\_tx\_0001.pcap
  - 20120824\_1650\_ff\_0000\_209A\_wifi1\_rx\_0001.pcap
  - 20120824\_1650\_ff\_0000\_209A\_wifi1\_tx\_0001.pcap
- Only Direction enabled
  - 20120824\_1650\_ff\_0000\_209A\_rx\_0001.pcap
  - 20120824\_1650\_ff\_0000\_209A\_tx\_0001.pcap
- Only Interface Enabled
  - 20120824\_1650\_ff\_0000\_209A\_wifi0\_0001.pcap
  - 20120824\_1650\_ff\_0000\_209A\_wifi1\_0001.pcap
- Both Direction & interface separation disabled
  - 20120806\_1310\_ff\_0000\_2068\_0001.pcap

Log file size can configured as below for wireless interfaces in MegaBytes(MB)'s:

```
Prompt#cli
#config log radio-logfilesize 40
#reboot
```

By default radio-logfilesize configured to 10 MegaBytes(MB)

The log files are copied to the USB storage (*/tmp/usb/ModelDeploymentPktCaptures*) for the size configured in database as above. This logging feature is very useful when we have multiple WSMP applications running and we require logging of all the packets.

## 5.3 Ethernet packet logging

The firmware supports logging all packets at the wired interface layer, which can be enabled through cli.

To check the status of logging, log on to board using telnet and issue login id and password

Execute “cli” command

Execute “show configuration”

Under syslog information check whether Ethernet is enabled

By default ethernet(under syslog) is disabled, for RSU

### 5.3.1 Configure Ethernet logging enable/disable:

To enable ethernetlogging:

Prompt# cli

Prompt# config log ethernetlog enable

Prompt# reboot

### 5.3.2 Configure Ethernet logging interface name:

Wired interface name should be configured as below to log Ethernet packets on that interface:

Prompt# cli

Prompt# config log loginterface brtrunk

Prompt# reboot

### 5.3.3 Configure Ethernet log separation by direction:

Prompt#cli

Prompt# apphalt

Prompt #config log ethernetlog-perdirection enable

Prompt#exit

### 5.3.4 Configure Ethernet log file size limit

Log file size can configured as below for wireless interfaces in MegaBytes(MB)’s:

Prompt#cli

#config log ethernet-logfilesize <value>

// Where, <value> is in units of MB, with float value indicating KB

// <value> as 10 is taken as (10\*1048576) bytes = 10MB

// <value> as 0.4 indicates (0.125 \* 1048576) bytes = 128KB

Prompt#exit

By default ethernet-logfilesize configured to 10 MegaBytes(MB)

### 5.3.5 Configure Ethernet log time limit

Configures the time limit for each file created, with default time log disabled.

```
Prompt#cli
Prompt# apphalt
Prompt#config log ethernet- logfiletime <value>
 // Where, <value> is in units of minutes
 // <value> as 10 is taken as ten minutes

Prompt#exit
```

Example file names:

1. Direction enabled
   
     20120824\_1650\_brtrunk\_0000\_209A\_rx\_0001.pcap
   
     20120824\_1650\_brtrunk\_0000\_209A\_tx\_0001.pcap
2. Direction disabled
   
     20120824\_1650\_brtrunk\_0000\_209A\_0001.pcap

By default following ethernet log options are disabled:

```
#config log ethernetlog disable
#config log ethernetlog-perdirection disable
```

The log files are copied to the USB storage (*/tmp/usb/ModelDeploymentPktCaptures*) for the size configured in database as above.

NOTE: Ethernet log application does not log packets on port 23 & 22.

## 5.4 Log file flush control

Both wirelss & Ethernet log files are copied to the USB storage (*/tmp/usb/ModelDeploymentPktCaptures*) after reaching configured size in database as mentioned in above sections and also log files can be flushed to USB storage as per every “*UploadTime*” mentioned in *logoffload.conf* even before reaching configured log file size by enabling the “log-file-flush” in database as below:

To enable log-file-flush:

```
Prompt#cli
#config log log-file-flush enable
#exit
```

This feature can be enabled and disabled run time without reboot.

**NOTE:** The changes in following parameters through cli will restart syslog:

```
#config log ethernet-logfilesize
#config log ethernetlog
#config log ethernetlog-perdirection
#config log interface
#config log log-file-flush
#config log radio-logfilesize
#config log radio-perdirection
#config log radio-perinterface
#config log radio-receivelog
#config log radio-transmitlog
#config log syslog-rotate-time
#config log syslog-server-ip
#config log syslog-server-port
#config log syslog-status
#config log time-based-syslog
```

## 5.5 Log files offload

### 5.5.1 Using firmware default keys

1. The private and public security keys are part of the locomate firmware, so no need to generate the security keys manually.
2. The private key is present in /root/ directory named scp\_key and the public key is /root/.ssh/authorized\_keys.
3. Execute the following statements, on Locomate :-  
Copy the public key to server as :-  
-> scp /root/.ssh/authorized\_keys [root@<server\\_ip>:scp\\_key.pub](#)
4. Execute the following statements, on Server:- If /root/.ssh/authorized\_keys does not exists then :-  
Copy scp\_key.pub to /root/.ssh/authorized\_keys as :-  
-> cp /root/scp\_key.pub /root/.ssh/authorized\_keys  
  
Change the permissions for /root/.ssh/authorized\_keys as :-  
-> chmod 600 /root/.ssh/authorized\_keys  
  
If /root/.ssh/authorized\_keys exists then:-  
Append scp\_key.pub to /root/.ssh/authorized\_keys :-  
-> cat /root/scp\_key.pub >> /root/.ssh/authorized\_keys
5. On Locomate :-  
Test the scp using the private key (/root/scp\_key) as :-  
-> scp -i /root/scp\_key /var/log/messages root@<server\_ip>:  
/\* scp should be successful without prompt for password \*/

For log-offload application update the security key path to /root/scp\_key, and also update the server address in 'logoffload.conf'.

### 5.5.2 Generating new keys

Locomate has the ability to periodically offload log files to the server, and is started by default upon bootup. The offload daemon uses rsync and scp to transfer the files and so requires the following configuration setting for secured communication and without any user interaction.

1. Generate keys as follows:

Generate key for scp transmission using Public Key Authentication and apply at remote host

```

--> Telnet the Locomate-OBU board && go to /var/ directory
--> Generate private key using command:
 shell-prompt# dropbearkey -t rsa -f scp_key
--> Extract the Public key using following command:
 shell-prompt# dropbearkey -f scp_key -y | head -n2 | tail -n1
>scp_key.pub ,change permission of both to chmod 600 scp_key*
--> Scp the Public Key to remote host using below:
 shell-prompt# scp scp_key.pub root@192.168.100.4:/.ssh
--> On remote host copy scp_key.pub to .ssh/authorized_keys file
 append key if file .ssh/authorized_key already exists i.e. cat scp_key.pub >>
.ssh/authorized_keys
--> change permission of authorized_keys as 600, chmod 600 .ssh/authorized_keys

```

Off load application uses a configuration file present at /var/logoffload.conf

## 2. The configuration has the following fields:

```

#Arada RSE SCP File Transfer Configuration Paramater List file
Message File Format
Modified Date: 03/25/2012
Version: 0.1
#
#
Credentials
UserName=root
#
#Destination Host Server Address
ServerAddress=no-server
#
Security Key
SecurityKey=scp_key
#
Destination Host File Path till top level RSE-log directory
FilePath=/tmp/
#
Upload Time in seconds
UploadTime=3600

```

username as credential :- user name for your system

serveradd:- ipv4/ipv6 address of your system. for ipv6 use some global address for both board and server.

key path -> path on board where key can be found

file path-> top directory on your system before RSE-logs directory

uploadtime -> time interval for upload in seconds

unit name:- is name of rse or board (can be obtained with apname output in 'cli->show configuration')

For example: filepath with /tmp/ (top directory on your system before RSE-logs directory)

then

/tmp/RSE-logs/CML/ARADA/ for cml

/tmp/RSE-logs/SSL/ARADA/ for ssl

If the filepath is specified as './' then the log files will be transferred to the home directory of the login

unit name:- is name of rse or board (can be obtained with apname output in 'cli->show configuration')

3. Connect a PC/laptop to board to act as host server,  
create 2 directories on server

`<filepath>/RSE-logs/CML/<unit name>/` for cml

`<filepath>/RSE-logs/SSL/<unit name>/` for ssl

4. This application will be started automatically and uses the configuration as in  
logoffload.conf
5. To Run Manually :  
shell-prompt# `nohup auto_off_load -c /var/logoffload.conf &`

**Note:** After updating the /var/logoffload.conf, send as SIGUSR1 signal to auto\_offload process as below to take updated parameters effect without rebooting or apphalt/apprun.

Kill -USR1 <pid of auto\_offload>

## 6. Bluetooth Applications

We provide Bluetooth applications which collect Bluetooth device MAC addresses and send the WSMP encapsulated data over service channel.

usage: bluetoothtx [sch channel access <1 - alternating> <0 - continuous>] [TA channel ] [ TA channel interval <1- cch int> <2- sch int>]

Example command to start a Bluetooth MAC address service (PSID=5) in alternative channel mode of operation.

```
$prompt> bluetoothtx 1 172 2
```

Similary we developed an application to receive the WSMP encapsulated Bluetooth data over DSRC radio and transmit to a Android enabled phone, which displays the list of Bluetooth devices present in the transmitter region.

usage: bluetoothrx [user req type<1-auto> <2-unconditional> <3-none>] [imm access] [extended access] [channel <optional>] [PROVIDER MAC <optional>]

Example command to receive the data from the Bluetooth MAC address service provider

```
$prompt>bluetoothrx 1 0 0
```

Android application can be downloaded from the Android Market with name 'LocoMessage'.



## 7. Configuring LocoMate

### 7.1 Command Line Interface (CLI)

This chapter lists all the supported CLI commands. Throughout this document, CLI commands will be denoted using bold, computer font as in command. Parameters to the command will be denoted by non-bold computer font as in

`cli`

parameter. Any values which are needed by the parameters will be denoted in `<parameter>`.

CLI prompt can be obtained by

telnet to the board

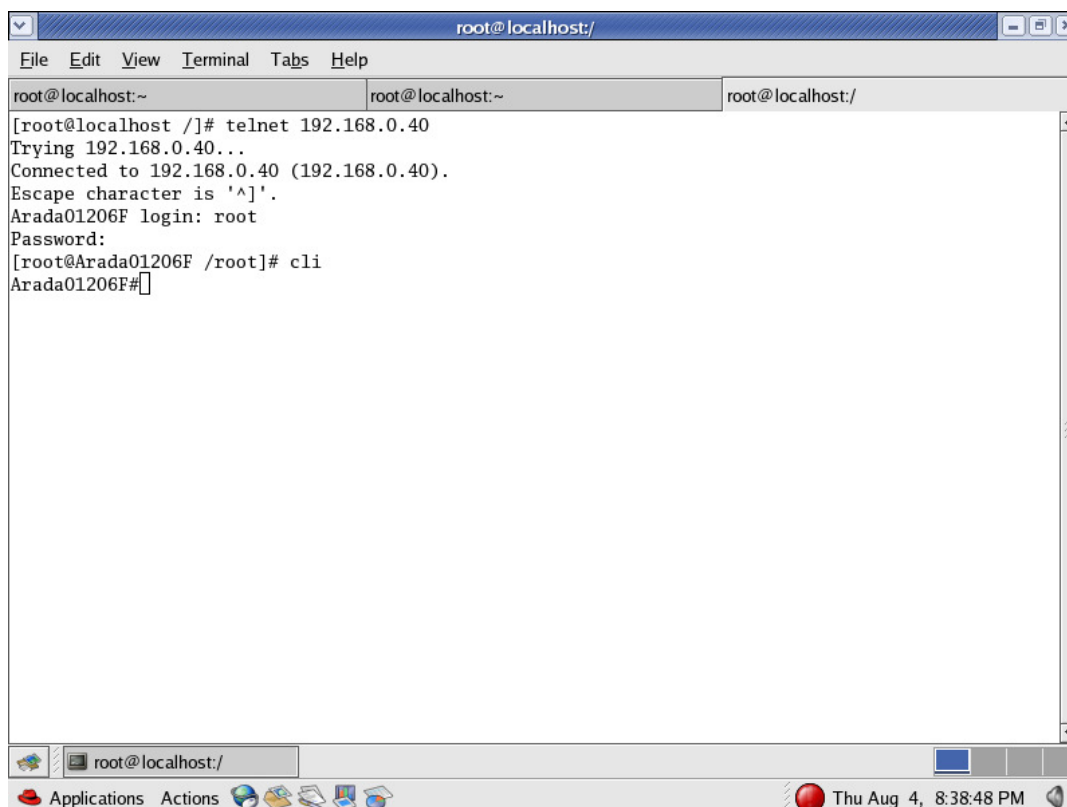
typing 'cli' at the linux prompt as in

```
$ cli
```

To go back to linux prompt ,type

```
$ exit
```

#### Screenshot with CLI prompt



### 7.2 Show configuration

At the CLI prompt enter the following commands:

```
Arada# show configuration
```

#### Screenshot with CLI show configuration output

```

root@localhost:~
File Edit View Terminal Tabs Help

 Not enabled
security-profile 3
 Not enabled
security-profile 4
 Not enabled

vlan
 management-vlan 1
 untagged-vlan-status enable
 untagged-vlan 1

snmp
 status enable
 trap-server-ip 0.0.0.0
 trap-server-community trap
 read-community public
 read-write-community private
 trap-port 162

applications
 application status Name Argument
application1 enable /usr/local/bin/getwsstxrxncdec -t\ BSM\ -o\ NORX\ -w\ Us
er\ -u\ 2\ -x\ 1\ -s\ 172\ -b\ 172\ -l\ /tmp/usb/\ -f\ pcap
application2 disable /usr/local/bin/getwsstxrxncdec -t\ BSM\ -o\ NORX\ -w\ Us
er\ -u\ 2\ -s\ 172\ -b\ 178\ -l\ /tmp/usb/\ -f\ pcap
application3 disable INVALID INVALID
application4 disable INVALID INVALID
Arada01206F#

```

## 7.3 Configuring AP name

We can change the name of the AP by

```
Arada# config apname <value>
```

## 7.4 Operational State change

We can perform a state transition to 'Halt' by executing 'apphalt' through CLI prompt. This command would kill all the DSRC applications, and enables change of any configuration to the same.

```
Arada# apphalt
```

*Note: Please perform 'apphalt', before attempting any configuration changes affecting the database or any input configuration files*

We can perform a state transition to 'Run' by executing 'apprun' through CLI prompt. This command would start all the DSRC applications, with user configured options.

```
Arada# apprun
```

## 7.5 Setting password

Password can be changed as mentioned below with combination of alpha numeric characters combined as upper and lower-cases including special characters.

```

#cli
> password <new password string>
#exit

```

Upon reconnection to unit, new password will take effect. Password string can take up to length 32.

## 7.6 Wireless MAC Address Randomization

We can configure to enable or disable wireless MAC address randomization through CLI prompt. This command will change the database entry for mac address randomization and applies upon powerup.

```
// To enable the mac address randomization upon powerup (default database value)
Arada# config mac-address-randomization enable
```

```
// To disable the mac address randomization upon powerup
Arada# config mac-address-randomization disable
```

## 7.7 Application profile creation

LocoMate provides configuring four application profiles with application name, arguments, and enable/disable status. The four applications are identified as 'application1', 'application2', 'application3' and 'application4'.

Application1 is configured and enabled to transmit BSM messages in channel 172 with radio operating in continuous channel mode.

Applicaiton2 is configured but disabled to transmit BSM messages in channel 178 with radio operating in alternative channel switch mode.

Application3 is not configured

Application4 is not configured

## 7.8 Switching between default configured applications

By default the Application1 is executing and to switch between Application1 and Applicaiton2, we need to just disable Application1 and Enable Application2 with the following commands at the CLI prompt.

```
Arada# config application app1Status disable
Arada# config application app2Status enable
```

**Screenshot of Application profile change commands**

```

root@localhost:~
File Edit View Terminal Tabs Help
root@localhost:~ root@localhost:~
vlan
 management-vlan 1
 untagged-vlan-status enable
 untagged-vlan 1

snmp
 status enable
 trap-server-ip 0.0.0.0
 trap-server-community trap
 read-community public
 read-write-community private
 trap-port 162

applications
 application status Name Argument
 application1 enable /usr/local/bin/getwsstxrncdec -t\ BSM\ -o\ NORX\ -w\ Us
er\ -u\ 2\ -x\ 1\ -s\ 172\ -b\ 172\ -l\ /tmp/usb/\ -f\ pcap
 application2 disable /usr/local/bin/getwsstxrncdec -t\ BSM\ -o\ NORX\ -w\ Us
er\ -u\ 2\ -s\ 172\ -b\ 178\ -l\ /tmp/usb/\ -f\ pcap
 application3 disable INVALID INVALID
 application4 disable INVALID INVALID
Arada01206F#config application app
app1Name app2Name app3Name app4Name
app1Status app2Status app3Status app4Status
applarg app2arg app3arg app4arg
Arada01206F#config application app1Status disable
Arada01206F#config application app2Status enable

```

## 7.9 Configuring a new Application to be executed by default

To configure a new application to execute by default, please see the below example to configure the application3 profile.

- We need to disable the application profile  
Arada# config application app3Status disable
- Input the application name  
Arada# config application app3Name <app name with complete path>  
The applications can be found on /usr/local/bin on LocoMate.
- Input the application arguments  
Arada# config application app3arg <arguments as given to execute from a telnet console>
- Enable the application  
Arada# config application app3Status enable

## 7.10 LCM Configuration

To configure the LCM (local certificate management) status, through CLI:

1) To disable LCM:

```

$ssh_prompt>cli
Arada# apphalt
Arada# config lcm-status disable
Arada# exit

```

2) To enable LCM :

```

$ssh_prompt>cli
Arada# apphalt
Arada# config lcm-status enable
Arada# exit

```

**NOTE:-** The default value is LCM enabled, for ASD devices.

## 7.11 WAAS Configuration

This feature has been moved to file based configuration, with values from file `'/var/gps.conf'` loaded at bootup time. WAAS feature is enabled by default. Sample configuration file is as in D.4.

## 7.12 Configure heartbeat message transmission

The device has the capability to transmit heartbeat messages as per the configuration file. This feature can be enabled with:

Arada# config ip traffic-control-server-connection-check enable

The default configuration file is present at /var directory, with filename – heartbeat.conf. Heartbeat messages are sent as per the ip address (ipv4 or ipv6) and port in the configuration file with required periodicity.

## 7.13 Startup script commands

In the above we have provided mechanism to create multiple application profiles to execute multiple applications at startup. It may be necessary to include any other custom applications to be executed at startup and users have option to include the same.

The applications can be found on `/usr/local/bin` on LocoMate.

To set up a new application as default start-up:

1. \$ vi /var/appstart

change this to application to one of the applications from `/usr/local/bin`.

Example `/usr/local/bin/getwbsstxrrencdec &`

The above command will set `getwbsstxrrencdec` as the default start-up application.

2. Save the file, quit vi editor and reboot.

## 7.14 Network configuration

Locomate has the following MAC layer devices:

- Bridge (brtrunk)
- Bridge (brwifi) – Configurable
- Eth0 – Ethernet
- Eth1 – Ethernet (not exposed for RSU)
- Wifi0vap0 – Radio 1
- Wifi1vap0 – Radio 2 (available for a dual radio SKU)

The architecture of a single bridge device is:

- Brtrunk (with the following interfaces)
  - eth0
  - eth1
  - wifi0vap0 - available only if it's a dual radio device
  - wifi1vap0 - available only if it's a dual radio device

The architecture of a dual bridge device with dual radio can be as:

- brtrunk (with the following interfaces)
  - eth0
  - eth1

- wifi0vap0
- brwifi (with the following interfaces)
  - wifi1vap0 - available only if it's a dual radio device

The architecture of a dual bridge device with single radio can be as:

- brtrunk (with the following interfaces)
  - eth0
  - eth1
- brwifi (with the following interfaces)
  - wifi0vap0

We could configure the following:

- ip addresses for the bridge, including ipv4 and ipv6
- configure the number of bridge interfaces
- configure the wifi interface to be attached to second bridge

### 7.14.1 Configure number of bridges

At the CLI prompt, enter the following commands:

```
Arada#config ip no-of-bridges [value] /* only 1 or 2 is
accepted */
Arada#exit (type exit to leave CLI prompt)
```

The values get in effect only upon reboot

### 7.14.2 Configure interface for the second bridge

At the CLI prompt, enter the following commands:

```
Arada# config ip interface-for-second-bridge [interface_name]
/*"wifi0vap0" or "wifi1vap0" or "both" */
Arada#exit (type exit to leave CLI prompt)
```

The values get in effect only upon reboot

### 7.14.3 Updating IPv4 Address of LocoMate

At the CLI prompt, enter the following commands:

```
Arada#config ip address <ip address> <netmask>
Arada#exit (type exit to leave CLI prompt)
```

Presently this is application for brtrunk

### 7.14.4 Updating IPv6 Addresses of brtrunk

At the CLI prompt, enter the following commands:

```
Arada# config ip ipv6Status [static/dynamic]
Arada# config ip ipv6-address ipv6value [prefix]
Arada# config ip ipv6-gateway-and-network-prefix [ipv6gateway
ipv6network-prefix prefixlen]
Arada#exit (type exit to leave CLI prompt)
```

### 7.14.5 Updating IPv6 Addresses of brwifi

This configuration is valid only if number of bridges is 2

At the CLI prompt, enter the following commands:

```
Arada# config ip brwifi-ipv6Status [static/dynamic]
```

```

Arada# config ip brwifi-ipv6-address ipv6value [prefix]
Arada# config ip brwifi-ipv6-gateway-and-network-prefix
[ipv6gateway ipv6network-prefix prefixlen]
Arada#exit (type exit to leave CLI prompt)

```

## 7.14.6 Creating ipv4-to-ipv6 tunnel

To enable and disable tunnel status :-

```
Arada #config ip tunnel-status[enable/disable]
```

To configure name, remote-ipv4 & tunnel-ipv6 :-

```
Arada #config ip tunnel-name-remote_ipv4-local_ipv6[name
remote_tunnel_ipv4 tunnel_local_ipv6]
```

To configure gateway and network prefix :-

```
Arada #config ip tunnel-ipv6-gateway-and-network-prefix[ipv6_gw
ipv6_net_prefix prefix_len]
```

## 7.15 Restore factory Default of LocoMate

At the CLI prompt, enter the following commands:

```
Arada#restore-factory-default
```

(LocoMate will reboot after restoring factory defaults and ip address will be set to 192.168.0.40)

## 7.16 Restore factory Default of LocoMate through udp

The command to issue to locomate with ip address '192.168.0.40' is

```
$prompt> echo " restore-configuration /etc/default-config" | nc -u 192.168.0.40 4444
```

(or)

```
$prompt> echo " restore-configuration /etc/default-config" | nc -u 192.168.0.40 5555
```

The ip address should be changed accordingly, or use broadcast address based on tool capability

## 7.17 Rebooting LocoMate from CLI

At the CLI prompt, enter the following commands:

```
Arada#reboot
```

**NOTE:** “apphlt” is required before “reboot” command as below:

```
Prompt# cli
```

```
Prompt# apphlt
```

```
Prompt# reboot
```

## 7.18 Backup Configuration

We maintain a database of the configuration on the device, which can help bringup the unit to same configuration after every power cycle. We can take backup of the configuration with the following, and requires setup of ftp server on the host machine.

Enter CLI prompt

```
Arada# backup-configuration <host server ip> <username> <password> <filename>
```

```
<host server ip> : IP address of the ftp server
```

```
<username> : ftp account username
```

```
<password> : ftp account password
```

<filename> : filename to which the database file (/var/config) is copied

## 7.19 Restore Configuration

We can restore the configuration from a file using the following command:

Enter CLI prompt

Arada# restore-configuration <host server ip> <username> <password> <filename>

<host server ip> : IP address of the ftp server

<username> : ftp account username

<password> : ftp account password

<filename> : filename which has the database information, and will be used restore the data based file (/var/config).



## 8. Upgrading LocoMate software

This section explains the steps to upgrade your firmware to a new version.

Always issue *Halt* command ‘*apphalt*’, through CLI, as in section 7.4 before starting the transfer of firmware and starting firmware upgrade process.

### 8.1 Firmware Contents

Firmware Package contains the following files

- **kernel.entrypoint<x.xx>** ← kernel entry point Address for version x.xx
- **kernel.md5** ← MD5 sum of the vmlinux file
- **root\_fs.md5** ← MD5 sum of the rootfs file
- **rootfs.squashfs** ← rootfs filesystem
- **vmlinux.bin.17** ← linux kernel image
- **WAVE\_LOCOMATE-200\_<x.xx>\_firmware.tar** ← Firmware tar file version x.xx

### 8.2 Checking the release name (version)

- Telnet to LocoMate.
- Type at prompt `$uname -r`  
`2.6.23-WAVE_LOCOMATE-200_X.XX` where x.xx will be the version number.

### 8.3 Upgrading the firmware using SCP

Always issue *Halt* command ‘*apphalt*’, through CLI, as in section 7.4 before starting the transfer of firmware and starting firmware upgrade process.

On LocoMate telnet prompt, run the following command:

```
$ firmware-upgrade-scp <path of firmware-file> <server-ip-address> <username>
<username> is optional, by default <username> is “root”. Host PC should have enabled SCP requests.
```

### 8.4 Upgrading the firmware using FTP

Always issue *Halt* command ‘*apphalt*’, through CLI, as in section 7.4 before starting the transfer of firmware and starting firmware upgrade process.

On LocoMate telnet prompt, run the following commands:

```
$ firmware-upgrade-ftp <username> <password> <Remote Host IP> <firmware tar file>
```

Configure the FTP server as per the host system used and create an username/password to be used for ftp transfer.

## 8.5 Upgrading the firmware using file on locomate

Always issue *Halt* command '*apphalt*', through CLI, as in section 7.4 before starting the transfer of firmware and starting firmware upgrade process.

Copy the firmware file to Locomate /tmp directory, using scp or any other transfer mechanism

On LocoMate telnet prompt, run the following commands:

```
$ cd /tmp;
$ nohup firmware-upgrade-file <firmware tar file> > /tmp/nohup.out
```

## 8.6 Configuring TFTP server

The procedure for upgrading to a new release version through the present executing version is as follows. In the procedure detailed here, it is assumed that LocoMate has the IP address as 192.168.0.40 and the tftp server IP address as 192.168.0.100.

1. Ensure that the IP addresses of LocoMate and tftp server with the firmware files are on the same network domain.
2. Start tftp server, with proper tftp server root directory (e.g /tftpboot)
3. Ensure that the new firmware files that you need to upgrade are located at /tftpboot or in the corresponding tftp server root directory.

## 8.7 Upgrading the firmware using TFTP

On LocoMate telnet prompt, run the following commands:

```
$ firmware-upgrade-tftp <firmware tar file> <tftp-server ip address>
```

Note: The upgrade process is expected to take 2-4mins, in the following sequence:

1. The power (lower) led blink while firmware upgrade process is going on, which should take around 2-3mins.
2. The board should then be automatically rebooted
3. WLAN led should glow
  - WLAN led will blink while transmit is in progress (see section 2.7 for more details)

Please find below sample screenshot of the messages while firmware upgrade is in progress:

(Debug messages are as per the latest firmware, so you may not find all the debug messages if you are presently having firmware 1.66 or older)

```
[root@Arada011807 /root]# firmware-upgrade-tftp 09512_ip_service_app_firmware.tar 192.168.0.111
*** Getting firmware file 09512_ip_service_app_firmware.tar using tftp server 192.168.0.111
*** Upgrading the firmware (power led should be blinking for 3-4 minutes)....
*** The device will reboot after upgrade completes
*** Please do not power off till the POWER LED blink stops....
*** If POWER LED blink continues for more than 5 minutes then firmware-upgrade failed....
Connection closed by foreign host.
[root@localhost ~]# ~
```

## 8.8 Updating the Security Certificate files through TFTP

Configure the tftp server on a host and place the *tar'd* certificate files in the tftp server root directory (which is typically /tftpboot). And, on LocoMate telnet prompt, run the following commands:

`$ seckey_update_tftp <certificates tar file> <tftp-server ip address>`

## 8.9 Updating Active message configuration files

An application named 'actmsglist' is used to manage the active message files. The details are as follows:

*actmsglist: is used to support addition/deletion/on-load/off-load features of active message list. All the active message list files are stored in /var/AML.*

*-l: to List contents and review contents the existing active message files.*

*-d: To delete the given active message file from list*

*-i:IP address*

*-g: to up-load the active message list (.tar) or adding single active message file via scp/tftp. -i should be mandatory*

*-o: to off-load the entire list to external system. -i should be mandatory.*

*-p: Mentioned protocol is used to up-load/addition/off-load. SCP=0(default), TFTP=1.*

*-u: User name. this option is used only when -p is 1. default is root.*

*-a: configurable limit of active message files. if no of active message file crossed the mentioned limit then addition will not happen.*

Examples:

List: actmsglist -l

Delete: actmsglist -d <name>

On-load with SCP: actmsglist -i <IP> -g <.tar> -u <user> -p 0

On-load with TFTP: actmsglist -i <IP> -g <.tar> -p 1

Adding file SCP: actmsglist -i <IP> -g <file> -u <user> -p 0 -a <limit>

Adding file TFTP: actmsglist -i <IP> -g <file> -p 1 -a <limit>

Off-load with SCP: actmsglist -i <IP> -o <.tar> -u <user> -p 0

Off-load with TFTP: actmsglist -i <IP> -o <.tar> -p 1

Note: in case of SCP on-load/Addition/ file has to be placed under users home directory.

Use of tftp requires a tftp server to be configured on the host machine.

## 8.10 Updating Heartbeat message configuration file through TFTP

Configure the tftp server on a host and place heartbeat conf file in the tftp server root directory (which is typically /tftpboot). And, on LocoMate telnet prompt, run the following commands:

`$ heartbeatconf_update_tftp /var/<filename> <tftp-server ip address>`

## 8.11 Updating logoffload configuration file through TFTP

Configure the tftp server on a host and place logoffload conf file in the tftp server root directory (which is typically /tftpboot). And, on LocoMate telnet prompt, run the following commands:

`$ logoffload_update_tftp /var/<filename> <tftp-server ip address>`

## 8.12 LocoMate TFTP Commands

LocoMate has a tftp client and can be used to transfer the files from LocoMate to Host or Host to LocoMate. Transferring files from LocoMate to Host requires the command in the following format:

```
[root@Arada334455 /root]# tftp --help
BusyBox v1.11.0 (2011-08-18 23:04:10 IST) multi-call binary

Usage: tftp [OPTION]... HOST [PORT]

Transfer a file from/to tftp server
Options:
 -l FILE Local FILE
 -r FILE Remote FILE
 -g Get file
 -p Put file
 -b SIZE Transfer blocks of SIZE octets
```

Example to transfer a file from Host to LocoMate

```
$prompt> tftp -g -r <filename on the host> <tftp server host ip address>
```

Example to transfer a file from LocoMate to Host

```
$prompt> tftp -p -l <filename on LocoMate> -r <filename on the host> <tftp server host ip address>
```

Note: Few *tftp* servers may require to be configured to enable creation of files on the host tftp server.

## 8.13 LocoMate FTP Commands

LocoMate has a *ftp* client and can be used to transfer the files from LocoMate to Host or Host to LocoMate. Locomate support gigabit Ethernet for faster file transfer. Transferring files from LocoMate to Host requires the command in the following format:

```
[root@Arada334455 /root]# ftpput
BusyBox v1.11.0 (2011-12-01 15:05:13 IST) multi-call binary

Usage: ftpput [options] remote-host remote-file local-file

Store a local file on a remote machine via FTP

Options:
 -v,--verbose Verbose
 -u,--username Username
 -p,--password Password
 -P,--port Port number
```

Example to transfer a file from LocoMate to Host, using 'test' user with password = 'test123'.

```
$prompt> ftpput -user test -p test123 <ftp server host ip address> <filename on the host> <filename on LocoMate>
```

Note: Few *ftp* servers may require to be configured to enable creation of files on the host ftp server.

## 8.14 Upgrading Firmware through Redboot

---

**Note:** We do not recommend this method for upgrading your firmware. Use this method only when Linux does not boot.

We recommend that you upgrade your firmware by using the tftp command described in Section 8.7, "Upgrading Firmware Using tftp"

---

To upgrade the firmware by reflashing the image, perform the procedure in the  
ARADA SYSTEMS CONFIDENTIAL

following sections:

## 8.14.1 Configuring the ip address at redboot

1. Configure the tftp server as in Section 8.3
2. Extract the firmware release to tftp directory. ex: '/tftpboot'
3. Connect host Ethernet to LocoMate
4. Configure host Ethernet IP address to 192.168.0.100
5. Set local IP address and host server IP address with:

```
RedBoot>i -l 192.168.0.40 -h 192.168.0.100
```

## 8.14.2 Update the Flash

This section explains steps to update the flash.

### 8.14.2.1.1 Updating the Linux kernel on the flash

```
RedBoot> load -r -b 0x80500000 vmlinux.bin.17
```

```
RedBoot> fis create -b 0x80500000 -l 0x00200000 -f 0xBF050000 -e 0x80331000 -r 0x80060000 vmlinux
```

**Note:** in the above line the value for **-e** option (entry point) should be checked/updated from the Firmware Package **kernel.entrypoint<x.xx>** file.

### 8.14.2.1.2 Updating the Linux file system on the flash

```
RedBoot> load -r -b 0x80500000 rootfs.squashfs
```

```
RedBoot> fis create -b 0x80500000 -l 0x00800000 -f 0xBF250000 -e 0x00000000 -r 0x00000000 rootfs
```

## 9. Using WaveDemo

WaveDemo is an application that enables you to access the features of the WAVE stack. [Figure 1.3](#) shows the steps to set up WaveDemo to transmit and receive data. To start a provider/user with channel switch and enable IP traffic flow through the service channel, you must register a provider application on one LocoMate device and register a user on another LocoMate device. Registration of an application enables the application to interact with the WSMP layer of the stack.

To start using WaveDemo you can arrange for a sample set up of two LocoMate devices. Turn on one LocoMate. This device runs the transmitting application, *getwbsstxrrencdec* by default. On the other LocoMate, start *wsmptdemo* and register a user with PSID = 32 (Hex value 0x20), to view receiving packet count.

The following subsections describe the steps in more detail. You can connect and access WaveDemo through telnet or through a remote host.

---

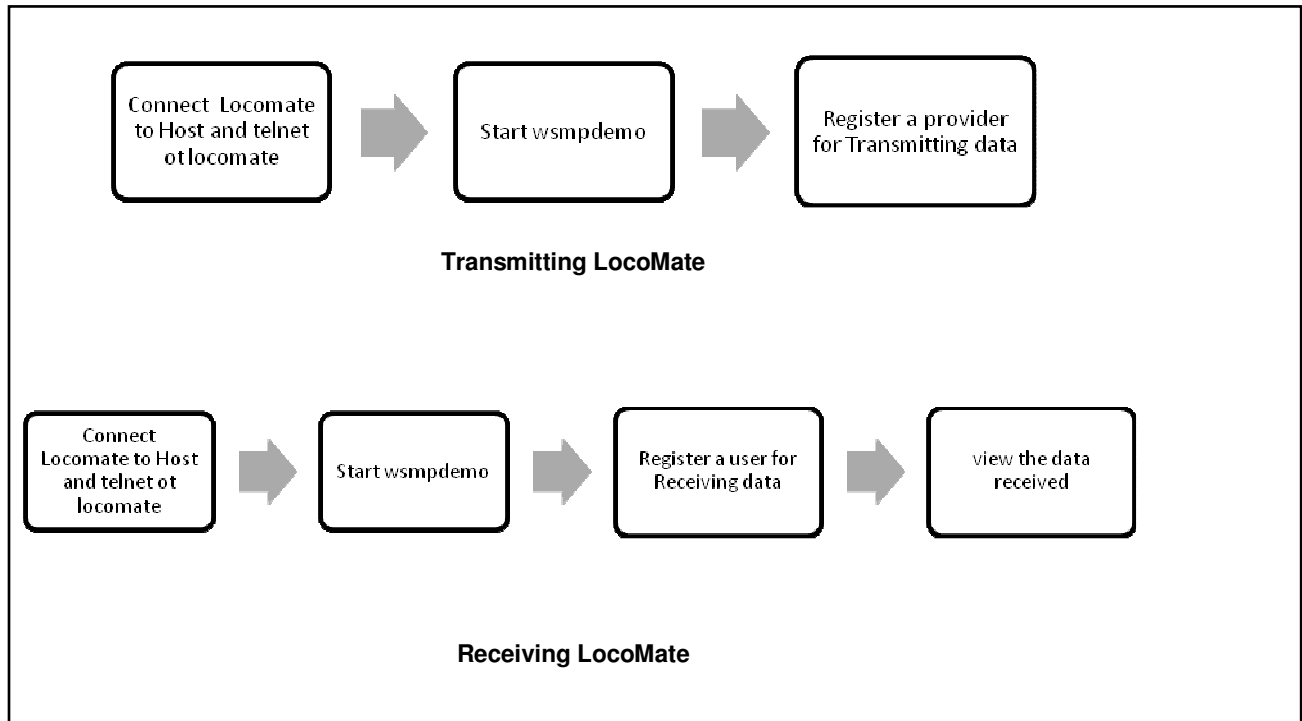
---

**Note:**

As *getwbsstxrrencdec* executes by default, LocoMate transmits WSMP encapsulated BSM data by default. To view data about receiving packets, you must first kill *getwbsstxrrencdec* and then run *wsmptdemo* as detailed below.

---

**Figure 1.3 Using WaveDemo**



#### Using waveDemo To Transmit and Receive data

### 9.1.1 Setting up a User for Receiving Data

To set up a user

1. Run wsmppdemo. The Startup screen appears.

**Figure 1.4 WaveDemo Startup Screen**

```

root@DP0013:~

* ARADA System's WAVE Demo Application v3.23 *

* <MAIN MENU> * <WSM PACKET> *
* * * * *
* [A] APPLICATION * Version:0 *
* [B] WBSS * Security:0 *
* [W] WSMP * Channel:0 *
* [R] WRSS * Rate:NotSet *
* [T] Get TSF Timer * Tx Power:0 *
* [I] CancelTX:ACI * PSID:0 *
* [C] CancelTX:Channel * * *
* [E] CANCEL TX * * *
* [G] Get Param * * *
* [S] Set Param * WSM Data(Len= 0) *
* [L] Log Messages * * *
* [/] Refresh Board * * *
* * * * *
* [Q] Quit * SRC=00:00:00:00:00:00*

* <WSMP Statistics> *
* Transmitted: 0 * Received: 0 *
* Bytes: 0 * Bytes: 0 *

* * *

```

3. Enter A. The **Application Registration Form** appears.
4. Enter U. The **User Registration Form** appears.
5. Select A User request type input, and enter 'y' for change channel to 172, and again enter 'y' for change of mode of operation to continuous, value = 1.

**Figure 1.5 User Registration page with 'UsrReqType' input**

```

root@DP0013:~
*
* [R] Reg. User * Version:0 *
* [U] UnReg. User * Security:0 *
* [Y] PSID * Channel:0 *
* [A] UsrReqType * Rate:NotSet *
* [C] ConfirmBeforeJoin * Tx Power:0 *
* [H] MatchAnyACM * PSID:0 *
* [I] Service IP *
* [S] Service Port *
*
* WSM Data(Len= 0) *
*
*
* [Q] Quit * SRC=00:00:00:00:00:00*

* <WSMP Statistics> *
* Transmitted: 0 * Received: 0 *
* Bytes: 0 * Bytes: 0 *

* 0:Alternative,1:Continues Viewed *

[Channel=172] Change (y/n)?:
New Value[0,255]:172
[0:Alternative,1:Continues=0] Change (y/n)?:
New Value[0,65535]:1

```

6. Select 'Y' to enter PSID, and input '11' as PSID value, and return to previous user registration screen

### PSID input screen



```

root@DP0013:~

* <APP-PSID MENU> * <WSM PACKET> *
* *
* [T] Traffic Control * Version:0 *
* [P] Private * Security:0 *
* [S] Public Safety * Channel:0 *
* [V] Vehicle Safety * Rate:NotSet *
* [I] Internet access * Tx Power:0 *
* [M] Security manager * PSID:0 *
* *
* *
* *
* *
* WSM Data(Len= 0) *
* *
* *
* [Q] Quit * SRC=00:00:00:00:00:00*

* <WSMP Statistics> *
* Transmitted: 0 * Received: 0 *
* Bytes: 0 * Bytes: 0 *

*

[ENTER 'y' THEN 1 FOR PRIVATE=0] Change (y/n)?:y
New Value[0,4294967295]:11

```

7. Select 'R' for registering the user

6. Press Esc to go to the main menu. The screen displays the data of the receive. The PSID of the rx packet is displayed and packet count increases over time.

**Figure 1.6** WSMP Rx Packet Update

```

root@DP0013:~

* ARADA System's WAVE Demo Application v3.23 *

* <APP-USER MENU> * <WSM PACKET> *
* *
* [R] Reg. User * Version:2 *
* [U] UnReg. User * Security:0 *
* [Y] PSID * Channel:172 *
* [A] UsrcReqType * Rate:3.0 *
* [C] ConfirmBeforeJoin * Tx Power:15 *
* [H] MatchAnyACM * PSID:11 *
* [I] Service IP *
* [S] Service Port *
* *
* *
* WSM Data(Len= 95) *
* 0] 0 *
* *
* *
* [Q] Quit * SRC=00:26:ad:01:17:fa*

* <WSMP Statistics> *
* Transmitted: 0 * Received: 327 *
* Bytes: 0 * Bytes: 34804 *

* (WSM) Received (104 Bytes) *

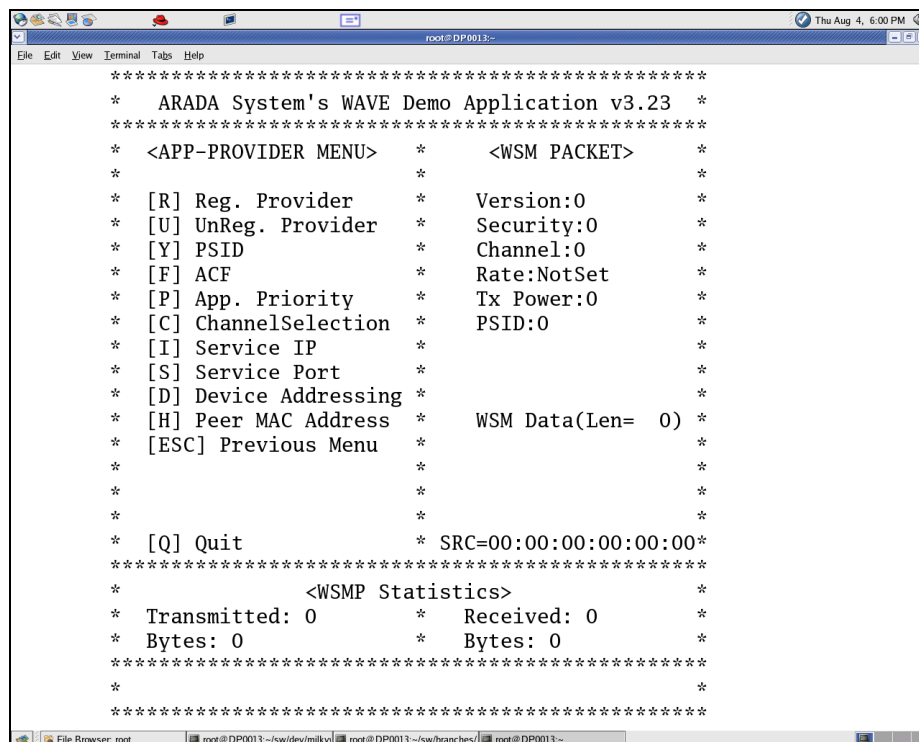
```

## 9.1.2 Setting Up a Provider for Transmitting Data

To set up a provider:

1. Run `wsmppdemo`. The Startup screen appears. Refer to [Section, “Using WaveDemo”](#) for more information.
2. Enter **A**. The **Application Registration Form** appears.
3. Enter **R**. The **Provider Registration Form** appears.

Figure 1.7 Provider Registration Form



```

* ARADA System's WAVE Demo Application v3.23 *

* <APP-PROVIDER MENU> * <WSM PACKET> *
* * * *
* [R] Reg. Provider * Version:0 *
* [U] UnReg. Provider * Security:0 *
* [Y] PSID * Channel:0 *
* [F] ACF * Rate:NotSet *
* [P] App. Priority * Tx Power:0 *
* [C] ChannelSelection * PSID:0 *
* [I] Service IP * *
* [S] Service Port * *
* [D] Device Addressing * *
* [H] Peer MAC Address * WSM Data(Len= 0) *
* [ESC] Previous Menu * *
* * * *
* [Q] Quit * SRC=00:00:00:00:00:00*

* <WSMP Statistics> *
* Transmitted: 0 * Received: 0 *
* Bytes: 0 * Bytes: 0 *


```

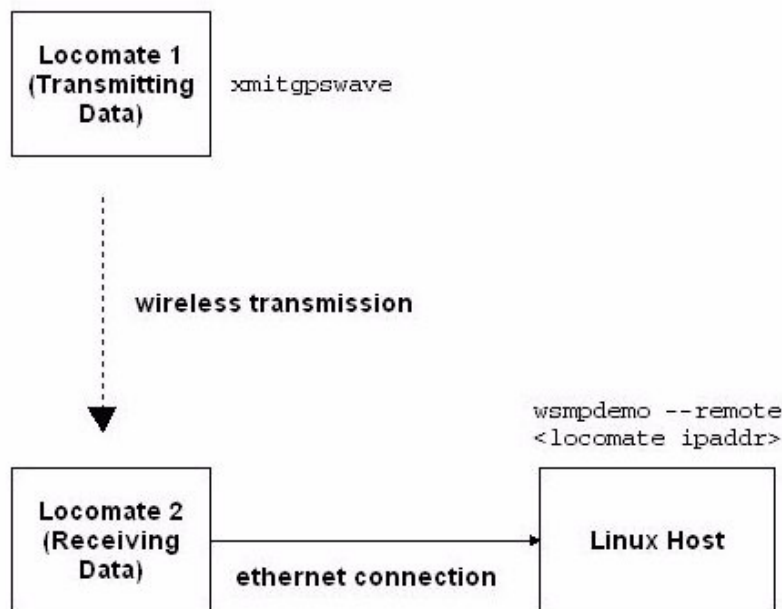
4. Enter **Y** to change the PSID.
5. Enter **C** to specify service channel.
6. Press **Esc** to get to the main menu.

## 9.2 Running WaveDemo from a Remote Host

By running `wsmptdemo` with the `--remote` parameter, you can configure a remote application to send and received WSMP packets. This command can also be run from a machine that does not have WAVE driver modules. Therefore, this option gives you more flexibility of using features of WaveDemo. You can register as user/provider to send/receive WSM Packets as well as use all the features that are available in the menu. You can also log GPS/wsmpt messages on the remote machine.

Figure 1.9

Remote Setup



To run WaveDemo from a Remote machine:

1. Connect LocoMate to the remote machine through the Ethernet port.

---

**Note:** The IP address of the remote machine must be in the same domain as the IP address of the LocoMate device. Requires IPV6 Address

---

2. Start WaveDemo by running the following command:

```
$prompt> wsmptdemo --remote <IPV6_of_LocoMate>
**This is IPV6 address of the Bridge interface brt.runk
```

**Example:**

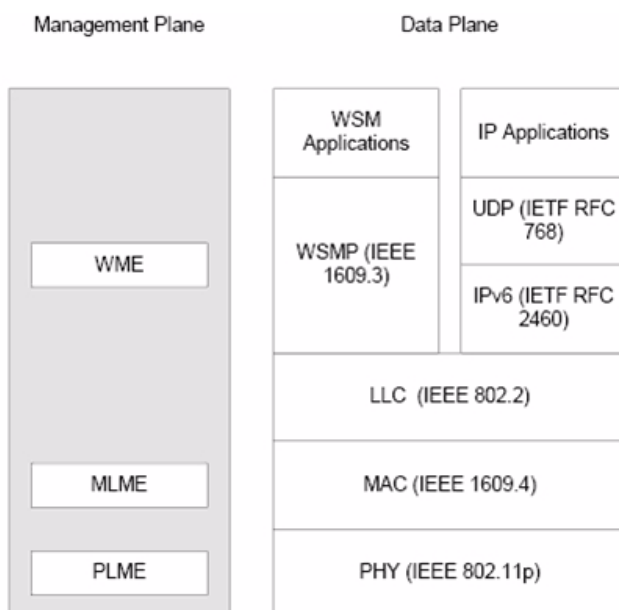
```
$prompt> wsmptdemo --remote fe80::200:ff:fe89:00
```

## A.1 Wave Radio Stack

## A.2 Overview of the Wave Radio Stack

Wireless Access in Vehicular Environments (WAVE) provides a communication protocol stack optimized for the vehicular environment, employing both customized and general-purpose elements as shown in [Figure A.1](#). WAVE Networking Services consists of the data plane's middle layers and most of the management plane (the WME).

**Figure A.1 Wave Radio Stack**



## A.3 Components of the Stack

The stack consists of the following layers:

- Data plane

This plane contains the communication protocols and physical layer used for data transfer. The data transfer occurs mainly between applications on both ends. It also carries traffic related to management plane entities.

- Management plane

This plane is responsible for system configuration and maintenance. Certain layers have their own Management Entities. For example, Physical Layer Management Entity (PLME), and MAC Layer Management Entity (MLME). The WAVE Management Entity (WME) is a general collection of management services.

## **B. IEEE 1609.1: WAVE Resource Manager**

It defines Resource Manager Application data read/write protocol between RSU and OBU.

### **B.1 IEEE 1609.2: 5.9 GHz Intelligent Transportation System (ITS) Radio Service Security**

It defines 5.9 GHz DSRC Security (formerly IEEE 1556) Anonymity, Authenticity and Confidentiality.

### **B.2 IEEE 1609.3: WAVE Networking Services**

It provides description and management of the DSRC Protocol Stack Application interfaces Network configuration management WAVE Short Message (WSM) transmission and reception.

### **B.3 IEEE 1609.4: WAVE Multi-Channel Operation**

It provides DSRC frequency band coordination and management Manages Lower Layer usage of the seven DSRC channels Integrates tightly with IEEE 802.11p.

### **B.4 IEEE 802.11p: Wireless LAN Medium Access Control (MAC) and physical layer (PHY) specifications: Wireless Access in Vehicular Environments (WAVE)**

It defines the Lower Layers of the communications stack Radio wave forms and wireless medium access procedures.

### **B.5 WAVE management entity (WME)**

The set of management functions, defined in IEEE 1609.3, required providing WAVE Networking Services; It is a management entity that performs these functions.

## **B.6 Wave Short Message Protocol (WSMP)**

The protocol is used for sending WAVE Short Messages. The WSMP allows applications to directly control physical characteristics, e.g., channel number and transmitter power, used in transmitting the messages. A sending application also provides the MAC address of the destination device, including the possibility of a broadcast address.

## C. Command Reference

### C.1 Using iwconfig

`iwconfig` is used to set parameters which are common across most drivers. For a detailed description of `iwconfig`, enter the following command:

```
man iwconfig
```

This section describes the use of `iwconfig` in the MADWiFi driver.

#### **iwconfig.help**

```
iwconfig --help
```

This command gives a brief help message.

#### **iwconfig.version**

```
iwconfig --version
```

This command returns the current version of `iwconfig` along with the version of the wireless extensions with which it was built.

#### **iwconfig [interface]**

```
iwconfig [interface]
```

This command returns the current wireless status of every network interface. If no interface is specified, the current wireless status of every network interface is returned. Non-wireless devices do not return any wireless status. `iwconfig interface [essid X ] [freq F ] [channel C ] [sens S ] [ap A ] [rate R ] [rts RT ] [frag FT ] [txpower T ] [enc E ] [key K ]`

This command enables you to change any of the optional parameters. Enter the parameters that you want to change.

#### **rate - Fix the Data Transmit Rate \***

This parameter is used to set the bit rate for cards supporting multiple bit rates. The value R is specified in bits/second and the values can be suffixed by k, M, or G for kilobits, megabits, and gigabits.

**Example** The following command sets the maximum bit rate to 36Mbps. Thus, the driver will automatically select the best rate less than or equal to 36Mbps.

```
myprompt# iwconfig ath0 rate 36M
```

**txpower - Set Transmit Power \***

This parameter sets the transmit power for data packets to W dBm. The value W can also be `auto` and `off` where `auto` uses automatic power control.

**Example** The following command sets all data packets to transmit at either 30 dBm or the maximum allowed in the current regulatory domain:

```
myprompt# iwconfig ath0 txpower 30
```

## C.2 Using wlanconfig

The current MADWiFi driver supports multiple APs and concurrent AP/Station mode operation on the same device. The devices are restricted to using the same underlying hardware, thus are limited to coexisting on the same channel and using the same physical layer features. Each instance of an AP or station is called a Virtual AP (or VAP). Each VAP can be in either AP mode, Station Mode, special station mode, or Monitor mode. Every VAP has an associated underlying base device which is created when the driver is loaded.

Creating and destroying VAP's are done through the `wlanconfig` tool found in the MADWiFi tools directory. Running the `wlanconfig` utility with no arguments returns a brief help line. The format of the `wlanconfig` command takes two forms:

```
wlanconfig VAP create wlandev Base Device wlanmode mode
[bssid k-bssid] [nosbeacon]
wlanconfig VAP destroy
```

Every Linux network device consists of a prefix followed by a number indicating the device number of the network device. For instance, the Ethernet devices are named `eth0`, `eth1`, `eth2`, and others. Each VAP which is created is also registered as a Linux network device. The value VAP can be either a prefix name of the Linux network device, or it can be the entire device name. For instance, specifying VAP as `ath` lets the Linux kernel add the network device as the next device with the prefix `ath`. Thus, the Linux kernel appends the proper number to the end to form the full device name, example, `ath1` if `ath0` already exists. However, the full device name can also be specified. For instance, VAP can also be `ath2`. In this case, the network device `ath2` is registered, regardless of whether `ath1` exists.

The Base Device is the underlying wireless network device name created when the driver is loaded. The MADWiFi driver creates `wifi0`, `wifi1`, and others as the underlying devices. By specifying the Base Device, the VAP is created with the Base Device as the parent device.

The mode is the operating mode of the VAP. The operating mode of the VAP cannot be changed once it is created.

In special cases, the operating mode of the VAP can be different from the operating mode of the underlying parent device. The first VAP which is created sets the operating mode of the underlying device. If the first VAP is deleted and a new VAP is created with a different operating mode than the original VAP, then the operating mode of the underlying device is changed to the new operating mode. The valid operating modes and their descriptions are given in [Table B.1](#).



**Example** Now, we wish to destroy a VAP (regardless of its operating mode). We assume there is a current VAP named `ath0` which is the one we wish to destroy.

```
myprompt# wlanconfig ath0 destroy
```

A new mode has been added, which is called `wavemode`.

```
wlanconfig ath0 create wlandev wifi0 wlanmode <Mode>
```

In this command, mode can also take the value: `wavemode`. This `wavemode` will create a 11p node.

To create a `wavemode` device, which sets the mode to WAVE mode (11p operation), run the following command:

```
wlanconfig ath0 create wlandev wifi0 wlanmode wavemode
```

## C.3 Private (non-standard) Driver Commands

The following is a list of the private commands which are accessible using `iwpriv`. The general syntax of `iwpriv` is:

```
iwpriv <device> [command] [parameters]
```

The entire list of `iwpriv` commands can be found by issuing an `iwpriv` to a device without any command.

```
iwpriv <device>
```

The resulting list of commands has several columns. The number of parameters allowed for each command is listed.

Parameters are classified as either set or get parameters. Set parameters are parameters which the user supplies to the driver. Get parameters are parameters which the driver returns to the user.

### List of wavemode iwpriv Commands

#### **get\_servicechan**

Number Input Arguments: 0

Number Returned Arguments: 1

Default value: N/A

Resets State Machine After Command: No This command gets current service channel value.

**Example** The following command gets `servicechan` value on `ath0` :

```
myprompt# iwpriv ath0 get_servicechan ath0
get_servicechan:172
```

#### **ipcchpermit**

Number Input Arguments: 1

Number Returned Arguments: 0

Default value: 0

Resets State Machine After Command: No

This command sets flag to allow IP traffic in control channel

**Example**        The following command sets value of ipcchpermit flag on ath0:

```
myprompt# iwpriv ath0 ipcchpermit 1
```

Enables the IP traffic in Control Channel

## **get\_ipcchpermit**

Number Input Arguments: 0

Number Returned Arguments: 1

Default value: 0

Resets State Machine After Command: No

**Example**        The following command gets value of ipcchpermit flag on ath0:

```
myprompt# iwpriv ath0 get_ipcchpermit
```

```
ath0 get_ipcchpermit :1
```

## **wsmppfillrssi**

### **wsmppfillrssi**

Number Input Arguments: 1

Number Returned Arguments: 0

Default value: 0

Resets State Machine After Command: No

This command sets flag to fill rssi value in the incoming WSMP traffic at 5'th byte of the body. This can be used by non-ASN encoded WSMP message applications, to provide the RSSI of every packet to the application layer. Since the data is modified, it is presently limited to the non-ASN encoded packets.

**Example**        The following command sets value of wsmppfillrssi flag on  
ath0:

```
myprompt# iwpriv ath0 wsmppfillrssi 1
```

enables the flag to set rssi on incoming wsmpp traffic

## **get\_wsmppfillrssi**

### **get\_wsmppfillrssi**

Number Input Arguments: 0

Number Returned Arguments: 1

Default value: 0

Resets State Machine After Command: No

This command gets current flag value of wsmppfillrssi

**Example**        The following command gets value of wsmppfillrssi flag on ath0:

```
myprompt# iwpriv ath0 get_wsmppfillrssi
```

```
ath0 get_wsmppfillrssi:0
```

## D. Sample configuration files

This chapter lists sample configuration files for active messages(store-and-forward) and immediate forward messages. The meaning of keywords is same across files, with some changes in the keyword names. For example 'MessageType' is same as 'Type'. Please see commented example file in the below sections, for meaning of each line.

### D.1 Active message configuration file (store-and-forward)

#### TIM data file to be present in Locomate

The following is an example configuration file present along with the firmware, at /var/AML/

```
Version=0.5
Type=TIM
PSID=0x8003
Priority=2
TxMode=ALT
TxChannel=178
TxInterval=1.0
DeliveryStart=04/15/2012, 00:00
DeliveryStop= 04/16/2012, 23:59
Signature=True
Encryption=False
Payload=30820108800110810900000000000000001000830101A481F03081ED800102A11BA119A0108
00418054A828104CE3582F582020CFD810200C0820102820207DB830306162184027D00850102A6
10800418052D158104CE3595AC82020CF68702016E880100A9663064800200C0A25EA05CA35A04
040636EB2B0404005FDEE00404F6A5C0CF0404007AF2E204040624F27904040A1DF4BB04040C6
1FAF504040E34FF5204040C160476040407EE0599040405BA07FE040404510A0C040400E809CC04
04FBE013D50404E395324DAA3AA0383006A0048002352A3006A0048002010C3006A00480023138
3006A004800222113006A0048002010C3006A004800231483006A0048002221185021001
```

The following is another example configuration file with informational comments present along with the firmware, at /var/AML/

```
Message File Format
Modified Date: 10/31/2011
Version: 0.1
#
Format Convention:
Comments are followed by either '#' or ';' and should not be
considered as part of the message items.
Empty lines should be ignored.
#
#
Message Dispatch Items
#
Message Type (e.g. TIM)
MessageType=TIM
#
Message PSID (as set by Test Conductor)
MessagePSID=0x8003

Message Priority (as set by Test Conductor) in the range of 0 through 7 with 0 being the lowest
priority
MessagePriority=2
#
Transmission Channel Mode (CONT or CCH/SCH as set by Test Conductor)
TransmissionMode=CCH/SCH
TransmissionChannel=178
#
Transmission Broadcast Interval 0.05 to 20.0 seconds for repeated broadcasts. A broadcast interval of
0 would indicate that the payload should be dispatched immediately and only one time.
TransmissionBroadcastInterval=0.1
#
Message Delivery (broadcast) start time (UTC date and time) in the form: "mm/dd/yyyy, hh:mm
MessageDeliveryStart=11/01/2011, 00:00
#
Message Delivery (broadcast) stop time (UTC date and time) in the form: "mm/dd/yyyy, hh:mm
MessageDeliveryStop= 11/30/2014, 23:59
#
Message Signature/Encryption
MessageSignature=True
MessageEncryption=False
#
Message Payload (encoded according to definition)
MessagePayload= 30 81 C0 80 01 10 81 09 00 00 00 00 00 00 00 10 00 83 01 01 A4 81 A8 30 81
A5 80 01 02 A1 1B A1 19 A0 10 80 04 19 24 C0 D4 81 04 CE 6A 5A 18 82 02 07 16 81 02 00 01 82
01 02 82 02 07 DB 83 03 06 16 21 84 02 7D 00 85 01 02 A6 10 80 04 19 24 E0 E6 81 04 CE 6A 52
20 82 02 07 13 87 02 04 4A 88 01 00 A9 1E 30 1C 80 02 00 02 A2 16 A0 14 A3 12 04 04 19 C0 1F
03 04 04 0B 12 19 7F 04 04 13 26 15 CE AA 3A A0 38 30 06 A0 04 80 02 35 29 30 06 A0 04 80 02
01 0C 30 06 A0 04 80 02 31 38 30 06 A0 04 80 02 22 11 30 06 A0 04 80 02 01 0C 30 06 A0 04 80
02 31 54 30 06 A0 04 80 02 22 11 85 02 10 01
```

## D.2 Immediate message forward configuration file

The following is an example configuration files expected by Locomate application over UDP in ascii form.

### MAP data file to be sent over UDP

```
Version=0.5
Type=MAP
PSID=0xBFF0
Priority=7
TxMode=CONT
TxChannel=172
TxInterval=0
DeliveryStart=
DeliveryStop=
Signature=True
Encryption=False
Payload=3081DE80011081090000000000000001000830101A481C63081C3800102A11BA119A01080
0418054A3B8104CE3585DF82020D0681020040820102820207DB830306162184027D00850102A61
080041804FD888104CE35C39E82020CF68702016E880100A93C303A80020040A234A032A3300404
1C6BCDB304040420EC2B0404FAC8EC280404EF79F1210404EBC4FD660404E65310690404F9621
AA50404095B3F31AA3AA0383006A004800235293006A0048002010C3006A004800231383006A00
4800222113006A0048002010C3006A004800231483006A0048002221185021001
```

### SPaT data file to be sent over UDP

```
Version=0.5
Type=SPAT
PSID=0xBFE0
Priority=7
TxMode=CONT
TxChannel=172
TxInterval=0
DeliveryStart=
DeliveryStop=
Signature=True
Encryption=False
Payload=3081C080011081090000000000000001000830101A481A83081A5800102A11BA119A010800
4181CB1B08104CCADAF0482020A0F81020060820102820207DB830306162084027D00850102A61
08004181C9B948104CCADCBEC82020A0F8702016E880100A91E301C80020030A216A014A31204
040821F81D040408CAFBE204040E57FF5AAA3AA0383006A0048002352A3006A0048002010C300
6A004800231233006A004800222113006A0048002010C3006A004800231463006A004800222118502
1001
```

## Sample script to send the configuration files

The below script is used to send the above configuration files over UDP to the Locomate, configured for ip address 192.168.0.40

```
#!/bin/bash

#run on time intervals
File=SPaT_R1.txt
IP=192.168.0.40
Port=4587
Rate=.1

for i in {1..10}
do
 clear
 cat $File | nc -u $IP $Port&
 sleep $Rate
done
killall -9 nc
clear
```

## D.3 ModelDeploymentRemovable configuration file

```
Vehicle Awareness Device
Configuration File Format
Modified Date: 04/25/2012
Version: 0.4
#
Format Convention:
Comments are followed by either '#' or ';' and should not be
considered as part of the configuration items.
Empty lines should be ignored.
#

Device Configuration Items
Model Deployment Device ID
Set by the Test Conductor at time of device installation.
Test Conductor.
Unprogrammed ID value = 0
Programmed ID value range: 0x0001 to 0xffff (1 to 65,535)
ModelDeploymentDeviceID=0

Fixed/Random TemporaryID value
Fixed two bytes with random two bytes TemporaryID, Full random TemporaryID
control flag.
If Fixed is chosen, the fixed value shall be the two high-order bytes of the
ModelDeploymentDeviceID assigned above
Fixed -high order two bytes (big endien notation) = 0, Random = 1
TemporaryIDControl=0

Memory Device Mount Time
Set by the device when the memory is first mounted by the device.
Units are UTC date and time
Unprogrammed value = 0
MemoryDeviceMountTimeDate=0

GPS Antenna Offset Values
Units follow J2375 definition for antenna offsets.
The offset value should be added to the value derived from the GPS
receiver to give the desired position value.
Units 0.01 meters (centimeters)
antOffsetX=0
antOffsetY=0
antOffsetZ=0

Vehicle Type Value
Allowed values defined in J2735
Not Equipped, Not known or unavailable = 0
VehicleType=0

Vehicle Size Values
Vehicle length and width units: 0.01 meters (centimeters)
Not known or unavailable = 0
VehicleLength=0
VehicleWidth=0
```

## D.4 GPS configuration file

```
GPS Configuration Parameter List File
#
Message File Format
Modified Date: 07/02/2013
Version: 0.2
##
Format Convention:
Comments are followed by either '#' or ';' and should not be
considered as part of the message items.
Empty lines should be ignored.
##
#
#GPS Device Name
DeviceName=/dev/ttyS0
#GPS Device Mode
4 For Automotive mode
0 For Portable mode
DeviceMode=4
#
#GPS Baud rate
BaudRate=115200
#
#GPS Data Update Rate[in msec]
UpdateRate=200
#WASS status
#=1 to enable WAAS
#=0 to disable WAAS
WAAS=1
#
#GSV Repeat Rate
GSVRepeatRate=5
#
#IP Address of gpsd server
IPAddress=127.0.0.1
#
#Port NO
PortNo=2947
```



## D.5 Locomate Safety Alert configuration file

```
Arada Locomate Configuration Parameters
#
#Time To Contact from Remote Vehicle in seconds
TTC=5
#Lane Width
LANEWIDTH=4.0
#message for Emergency Electronic Brake Light
EEBL=Alert-Adjacent Lane Vehicle Decelerating
#message for Curve Speed Warning
CSW=Alert-Curve Ahead
#message for Forward Collision Warning
FCW=Forward-collision warning
#display type[bluetooth/led/info], default is info
DISPLAY_TYPE=info
```

## E. LCM Files

### E.1 Sample ROOT cert file contents

```
02 ff 04 00 83 ff 01 00 01 00 04 13 97 09 dd 00 00 00 01 01 02 69 df 01 90 66 7e 3c bd ca d0 9d 70 33
cc ac 2f fc 0f f1 30 24 96 cc 92 f9 c6 e5 0b 9d b1 72 b4 02 00 02 93 53 54 7b ef 08 05 45 50 c0 a8 36 7f
9a dc c2 b2 3f e2 b5 b0 17 46 b3 ee a4 a8 69 b5 f2 eb 30 00 3e c9 34 3e fe f4 b9 d6 ce 32 4b 93 35 4c 3e
8b 38 37 c4 93 67 b7 01 b5 64 15 3b 53 59 f8 e2 17 9d 1f eb 3c 0c 34 bd c2 95 7c ac e8 c7 b3 3d 3a 9e
a1 dd 77 a7 3c 65 2e 07 26 4d a9 02 b9 5a 09
```

## E.2 Sample LCM Configuration File

```

#*
#* LCM sample configuration file
#*
#*
*****/
#
Network configuration parameters

Servers host name or ip address
#RA_ADDRESS=12.172.124.246
RA_ADDRESS=2001:1890:110e:a777::f234

Servers port number
RA_PORT=16092

PSID in decimal
PSID=32

Certificate storage (kb)
Storage_Space=20480

#
#Bootstrap
#
Seconds to wait for bootstrap confirmation before requesting again
Bootstrap_Request_Timeout=30

Certificate Request Params, for SuperBatch and SubBatch
Values: 0 = seconds, 1 = minutes, 2 = hours, 3= 60-hours, 4 = years
Superbatch_Duration_units=2
Superbatch_Duration=120
Batch_Duration_Units=2
Batch_Duration_Value=120

Minimum delay between Certificate Request Status inquiries (in secs)
Certificate_Request_Status_Inquiry_Interval=70

Seconds to wait for confirmation before requesting again (in secs)
Certificate_Request_Confirmation_Timeout=30

Minimum delay between decryption-key requests (in secs)
Decryption_Key_Request_Interval=5

Storage thresholds (in secs)
For Example:
1 yr x 365 days/yr x 24 hr/day x 60 min/hr x 60 sec/min = 31536000secs
Maximum_Certificate_Storage_Time=31536000

500 min. x 60 sec/min
Request_Certificates_Time=30000

250 min x 60 sec/min
Request_Decryption_Key_Time=15000

General
Name to include in the certificate
LCM_NAME=obe
Minimum delay between connection requests (in secs)
Connection_Retry_Interval=5

Logging options
Values: 0/1 Default: 0
Enable writting to the log
LCMLogEnable=1

Name of directory for log files
LogFileDirectory=/tmp/usb/lcm

Simple name: lcm.log
LogUseSimpleName=lcm.log

The following options enable additional debugging information
Include lcm specific log messages

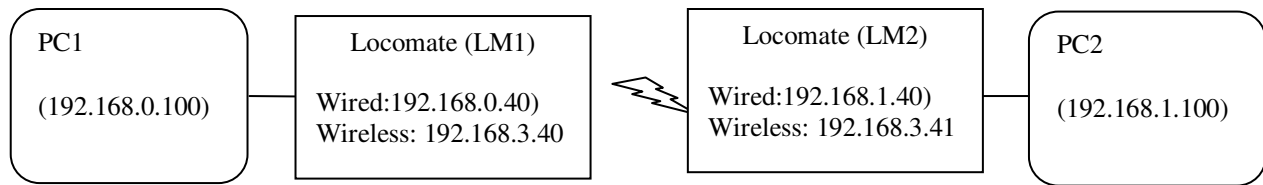
```

## E.3 Sample LCM log file

```
Feb 18 15:39:24 : logfile start
Feb 18 15:39:24 : LCMStatus File Not Found
Feb 18 15:39:39 : [BOOTSTRAP_REQ]:connect() failed.. errno=[128]
Feb 18 15:40:24 : [BOOTSTRAP_REQ]:connect() failed.. errno=[128]
Feb 18 15:41:09 : [BOOTSTRAP_REQ]:connect() failed.. errno=[128]
Feb 18 15:41:54 : [BOOTSTRAP_REQ]:connect() failed.. errno=[128]
Feb 18 15:42:39 : [BOOTSTRAP_REQ]:connect() failed.. errno=[128]
Feb 18 15:43:24 : [BOOTSTRAP_REQ]:connect() failed.. errno=[128]
Feb 18 15:44:35 : logfile start
Feb 18 15:44:35 : LCMStatus:0, Read Successfully
Feb 18 15:44:58 : logfile start
Feb 18 15:44:58 : LCMStatus:0, Read Successfully
Feb 18 15:45:13 : [BOOTSTRAP_REQ]:connect() failed.. errno=[128]
Feb 18 15:45:58 : [BOOTSTRAP_REQ]:connect() failed.. errno=[128]
Feb 18 15:46:43 : [BOOTSTRAP_REQ]:connect() failed.. errno=[128]
Feb 18 15:47:29 : logfile start
Feb 18 15:47:29 : LCMStatus:0, Read Successfully
Feb 18 15:47:53 : logfile start
Feb 18 15:47:53 : LCMStatus:0, Read Successfully
Feb 18 15:48:08 : [BOOTSTRAP_REQ]:connect() failed.. errno=[128]
Feb 18 15:48:47 : logfile start
Feb 18 15:48:47 : LCMStatus:0, Read Successfully
Feb 18 15:52:11 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 16:02:23 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 16:12:35 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 16:22:47 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 16:32:59 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 16:43:11 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 16:53:23 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 17:03:35 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 17:13:47 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 17:23:59 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 17:34:11 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 17:44:23 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 17:54:35 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 18:04:47 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 18:14:59 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 18:25:11 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 18:35:23 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 18:45:35 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 18:55:47 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 19:05:59 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 19:16:11 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 19:26:24 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 19:36:36 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 19:46:48 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 19:57:00 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 20:07:12 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 20:17:24 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 20:27:36 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 20:37:48 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
Feb 18 20:48:00 : [BOOTSTRAP_REQ]:connect() failed.. errno=[145]
```

## F. Configuration for point-to-point setup

The below procedure shows to setup a communication between two PCs through Locomate units, where PC can serve multiple purposes.



PC1: (192.168.0.100)

;Set the wired interface IP as 192.168.0.100 and gw as 192.168.0.40

On Windows, execute the following commands in command prompt, with command prompt executed with 'Run as Administrator'

```
route add 192.168.3.0 mask 255.255.255.0 192.168.0.40
route add 192.168.1.0 mask 255.255.255.0 192.168.0.40
```

>Output should be as follows:

```
>C:\Windows\system32>route add 192.168.3.0 mask 255.255.255.0 192.168.0.40
>Ok!
>
>C:\Windows\system32>route add 192.168.1.0 mask 255.255.255.0 192.168.0.40
>Ok!
```

; Disable firewall

;The two route entries can be placed in a batch file(rte.bat) and executed with 'run as administrator'

;for quick execution at every bootup

```

LM1:(192.168.0.40)
```

```

/var/appstart
```

```

#!/bin/sh
ifconfig brwifi 192.168.3.40 up
iwpriv wifi0vap0 ipcchpermit 1
echo 1 > /proc/sys/net/ipv4/conf/default/forwarding
echo 1 > /proc/sys/net/ipv4/conf/all/forwarding
echo 1 > /proc/sys/net/ipv4/conf/default/proxy_arp
```

```
echo 1 > /proc/sys/net/ipv4/conf/all/proxy_arp
route add -net 192.168.1.0 netmask 255.255.255.0 brwifi
```

-----

PC2:(192.168.1.100)

;Set the wired interface IP as 192.168.1.100 and gw as 192.168.1.40

On Windows, execute the following commands in command prompt, with command prompt executed with 'Run as

;Administrator'

```
route add 192.168.3.0 mask 255.255.255.0 192.168.1.40
route add 192.168.0.0 mask 255.255.255.0 192.168.1.40
```

>Output should be as follows:

>C:\Windows\system32>route add 192.168.3.0 mask 255.255.255.0 192.168.1.40

>Ok!

>

>C:\Windows\system32>route add 192.168.0.0 mask 255.255.255.0 192.168.1.40

>Ok!

; Disable firewall

;The two route entries can be placed in a batch file(rte.bat) and executed with 'run as administrator'

;for quick execution at every bootup

-----

LM2:(192.168.1.40)

-----

/var/appstart

-----

```
#!/bin/sh
ifconfig brwifi 192.168.3.41 up
iwpriv wifi0vap0 ipcchpermit 1
echo 1 > /proc/sys/net/ipv4/conf/default/forwarding
echo 1 > /proc/sys/net/ipv4/conf/all/forwarding
echo 1 > /proc/sys/net/ipv4/conf/default/proxy_arp
echo 1 > /proc/sys/net/ipv4/conf/all/proxy_arp
route add -net 192.168.0.0 netmask 255.255.255.0 brwifi
```

-----