SmartMesh WirelessHART Manager CLI Guide
<table>
<thead>
<tr>
<th>Section</th>
<th>Command</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.5.3</td>
<td>get alarms</td>
<td>46</td>
</tr>
<tr>
<td>3.5.4</td>
<td>get blacklist</td>
<td>47</td>
</tr>
<tr>
<td>3.5.5</td>
<td>get mote</td>
<td>48</td>
</tr>
<tr>
<td>3.5.6</td>
<td>get motes</td>
<td>51</td>
</tr>
<tr>
<td>3.5.7</td>
<td>get network</td>
<td>55</td>
</tr>
<tr>
<td>3.5.8</td>
<td>get otapfiles</td>
<td>58</td>
</tr>
<tr>
<td>3.5.9</td>
<td>get otapmotes</td>
<td>59</td>
</tr>
<tr>
<td>3.5.10</td>
<td>get otapstatus</td>
<td>60</td>
</tr>
<tr>
<td>3.5.11</td>
<td>get paths</td>
<td>61</td>
</tr>
<tr>
<td>3.5.12</td>
<td>get path</td>
<td>62</td>
</tr>
<tr>
<td>3.5.13</td>
<td>get redundancy</td>
<td>63</td>
</tr>
<tr>
<td>3.5.14</td>
<td>get security</td>
<td>64</td>
</tr>
<tr>
<td>3.5.15</td>
<td>get sla</td>
<td>65</td>
</tr>
<tr>
<td>3.5.16</td>
<td>get source route</td>
<td>66</td>
</tr>
<tr>
<td>3.5.17</td>
<td>get system</td>
<td>67</td>
</tr>
<tr>
<td>3.5.18</td>
<td>get users</td>
<td>68</td>
</tr>
<tr>
<td>3.6</td>
<td>help</td>
<td>69</td>
</tr>
<tr>
<td>3.7</td>
<td>logout</td>
<td>70</td>
</tr>
<tr>
<td>3.8</td>
<td>onechan</td>
<td>71</td>
</tr>
<tr>
<td>3.9</td>
<td>otap</td>
<td>72</td>
</tr>
<tr>
<td>3.9.1</td>
<td>otap start</td>
<td>72</td>
</tr>
<tr>
<td>3.9.2</td>
<td>otap status</td>
<td>73</td>
</tr>
<tr>
<td>3.9.3</td>
<td>otap stop</td>
<td>74</td>
</tr>
<tr>
<td>3.10</td>
<td>ping</td>
<td>75</td>
</tr>
<tr>
<td>3.10.1</td>
<td>ping show</td>
<td>77</td>
</tr>
<tr>
<td>3.10.2</td>
<td>ping stop</td>
<td>78</td>
</tr>
<tr>
<td>3.11</td>
<td>set</td>
<td>79</td>
</tr>
<tr>
<td>3.11.1</td>
<td>set acl</td>
<td>79</td>
</tr>
<tr>
<td>3.11.2</td>
<td>set blacklist</td>
<td>80</td>
</tr>
<tr>
<td>3.11.3</td>
<td>set mote</td>
<td>81</td>
</tr>
<tr>
<td>3.11.4</td>
<td>set network</td>
<td>83</td>
</tr>
<tr>
<td>3.11.5</td>
<td>set security</td>
<td>86</td>
</tr>
<tr>
<td>3.11.6</td>
<td>set sla</td>
<td>87</td>
</tr>
<tr>
<td>3.11.7</td>
<td>set system</td>
<td>88</td>
</tr>
<tr>
<td>3.11.8</td>
<td>set user</td>
<td>90</td>
</tr>
<tr>
<td>3.12</td>
<td>show</td>
<td>91</td>
</tr>
<tr>
<td>3.12.1</td>
<td>show cli-sessions</td>
<td>91</td>
</tr>
<tr>
<td>3.12.2</td>
<td>show events</td>
<td>92</td>
</tr>
<tr>
<td>3.12.3</td>
<td>show frame</td>
<td>93</td>
</tr>
<tr>
<td>3.12.4</td>
<td>show ap</td>
<td>95</td>
</tr>
<tr>
<td>3.12.5</td>
<td>show linksmap</td>
<td>97</td>
</tr>
<tr>
<td>3.12.6</td>
<td>show mote</td>
<td>98</td>
</tr>
<tr>
<td>3.12.7</td>
<td>show motes</td>
<td>100</td>
</tr>
<tr>
<td>Section</td>
<td>Title</td>
<td>Page</td>
</tr>
<tr>
<td>---------</td>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>3.12</td>
<td>Access Point CLI Commands</td>
<td></td>
</tr>
<tr>
<td>3.12.8</td>
<td>show msg</td>
<td>101</td>
</tr>
<tr>
<td>3.12.9</td>
<td>show optimization</td>
<td>102</td>
</tr>
<tr>
<td>3.12.10</td>
<td>show sessions</td>
<td>103</td>
</tr>
<tr>
<td>3.12.11</td>
<td>show stat</td>
<td>104</td>
</tr>
<tr>
<td>3.12.12</td>
<td>show status</td>
<td>110</td>
</tr>
<tr>
<td>3.12.13</td>
<td>show time</td>
<td>112</td>
</tr>
<tr>
<td>3.12.14</td>
<td>show ver</td>
<td>113</td>
</tr>
<tr>
<td>3.13</td>
<td>sm</td>
<td>114</td>
</tr>
<tr>
<td>3.14</td>
<td>su</td>
<td>116</td>
</tr>
<tr>
<td>3.15</td>
<td>trace</td>
<td>117</td>
</tr>
<tr>
<td>4</td>
<td>Access Point CLI Commands</td>
<td></td>
</tr>
<tr>
<td>4.1</td>
<td>Logging on to the Access Point CLI</td>
<td>120</td>
</tr>
<tr>
<td>4.2</td>
<td>radiotest start</td>
<td>121</td>
</tr>
<tr>
<td>4.3</td>
<td>radiotest stop</td>
<td>122</td>
</tr>
<tr>
<td>4.4</td>
<td>radiotest tx</td>
<td>123</td>
</tr>
<tr>
<td>4.5</td>
<td>radiotest set txpower</td>
<td>125</td>
</tr>
<tr>
<td>4.6</td>
<td>radiotest rx</td>
<td>126</td>
</tr>
<tr>
<td>4.7</td>
<td>radiotest getstats</td>
<td>127</td>
</tr>
<tr>
<td>4.8</td>
<td>show stats</td>
<td>128</td>
</tr>
<tr>
<td>4.9</td>
<td>show status</td>
<td>131</td>
</tr>
<tr>
<td>5</td>
<td>Managing Users and Passwords</td>
<td></td>
</tr>
<tr>
<td>5.1</td>
<td>Managing Users and Passwords for the Manager CLI</td>
<td>132</td>
</tr>
<tr>
<td>5.2</td>
<td>Changing the Linux Login Password</td>
<td>133</td>
</tr>
<tr>
<td>6</td>
<td>Linux Console Commands</td>
<td></td>
</tr>
<tr>
<td>6.1</td>
<td>Logging into the Linux console</td>
<td>134</td>
</tr>
<tr>
<td>6.2</td>
<td>clear-motes</td>
<td>134</td>
</tr>
<tr>
<td>6.3</td>
<td>config-login</td>
<td>135</td>
</tr>
<tr>
<td>6.4</td>
<td>config-ppp</td>
<td>136</td>
</tr>
<tr>
<td>6.5</td>
<td>create-network-snapshot</td>
<td>137</td>
</tr>
<tr>
<td>6.6</td>
<td>dust-manager</td>
<td>138</td>
</tr>
<tr>
<td>6.7</td>
<td>mgrctl</td>
<td>139</td>
</tr>
<tr>
<td>6.8</td>
<td>Network Configuration</td>
<td>140</td>
</tr>
<tr>
<td>6.9</td>
<td>restore-dcc-conf</td>
<td>141</td>
</tr>
<tr>
<td>6.9.1</td>
<td>restore-dcc-conf</td>
<td>141</td>
</tr>
<tr>
<td>6.10</td>
<td>restore-factory-conf</td>
<td>142</td>
</tr>
<tr>
<td>6.11</td>
<td>set-conf-param</td>
<td>143</td>
</tr>
<tr>
<td>6.11.1</td>
<td>set-conf-param</td>
<td>143</td>
</tr>
<tr>
<td>6.12</td>
<td>set-time</td>
<td>144</td>
</tr>
</tbody>
</table>
1 About This Guide

1.1 Related Documents

The following documents are available for the SmartMesh WirelessHART network:

Getting Started with a Starter Kit

- SmartMesh WirelessHART Easy Start Guide - walks you through basic installation and a few tests to make sure your network is working
- SmartMesh WirelessHART Tools Guide - the Installation section contains instructions for the installing the serial drivers and example programs used in the Easy Start Guide and other tutorials.

User Guide

- SmartMesh WirelessHART User's Guide - describes network concepts, and discusses how to drive mote and manager APIs to perform specific tasks, e.g. to send data or collect statistics. This document provides context for the API guides.

Interfaces for Interaction with a Device

- SmartMesh WirelessHART Manager CLI Guide - used for human interaction with a Manager (e.g. during development of a client, or for troubleshooting). This document covers connecting to the CLI and its command set.
- SmartMesh WirelessHART Manager API Guide - used for programmatic interaction with a manager. This document covers connecting to the API and its command set.
- SmartMesh WirelessHART Mote CLI Guide - used for human interaction with a mote (e.g. during development of a sensor application, or for troubleshooting). This document covers connecting to the CLI and its command set.
- SmartMesh WirelessHART Mote API Guide - used for programmatic interaction with a mote. This document covers connecting to the API and its command set.

Software Development Tools

- SmartMesh WirelessHART Tools Guide - describes the various evaluation and development support tools included in the SmartMesh SDK including tools for exercising mote and manager APIs and visualizing the network.

Application Notes

- SmartMesh WirelessHART Application Notes - app notes covering a wide range of topics specific to SmartMesh WirelessHART networks and topics that apply to SmartMesh networks in general.

Documents Useful When Starting a New Design
• The Datasheet for the LTC5800-WHM SoC, or one of the castellated modules based on it, or the backwards compatible LTP5900 22-pin module.
• The Datasheet for the LTP5903-WHR embedded manager.
• A Hardware Integration Guide for the mote SoC or castellated module, or the 22-pin module - this discusses best practices for integrating the SoC or module into your design.
• A Hardware Integration Guide for the embedded manager - this discusses best practices for integrating the embedded manager into your design.
• A Board Specific Integration Guide - For SoC motes and Managers. Discusses how to set default IO configuration and crystal calibration information via a “fuse table”.
• Hardware Integration Application Notes - contains an SoC design checklist, antenna selection guide, etc.
• The ESP Programmer Guide - a guide to the DC9010 Programmer Board and ESP software used to program firmware on a device.
• ESP software - used to program firmware images onto a mote or module.
• Fuse Table software - used to construct the fuse table as discussed in the Board Specific Integration Guide.

Other Useful Documents

• A glossary of wireless networking terms used in SmartMesh documentation can be found in the SmartMesh WirelessHART User’s Guide.
• A list of Frequently Asked Questions
1.2 Conventions Used

The following conventions are used in this document:

**Computer type** indicates information that you enter, such as specifying a URL.

**Bold type** indicates buttons, fields, menu commands, and device states and modes.

*Italic type* is used to introduce a new term, and to refer to APIs and their parameters.

- **Tips** provide useful information about the product.

- **Informational text** provides additional information for background and context.

- **Notes** provide more detailed information about concepts.

- **Warning!** Warnings advise you about actions that may cause loss of data, physical harm to the hardware or your person.

- code blocks display examples of code.
The CLI commands are described using the following notations and terminology:

| | Indicates alternatives for a field. For example, `<moteId> | #<MAC>` indicates that you can specify a mote by its mote ID or MAC address.
| < > | Indicates a required field.
| { } | Indicates a group of fields.
| [ ] | Indicates an optional field.

**MAC address**

When specifying a MAC address, do not use spaces. You may omit leading zeros and hyphens. In cases where the command syntax allows either the MAC address or mote ID to be specified, the MAC address must be preceded by the `#` symbol.

The following examples are all valid:

- `22CA`
- `00000000000022CA`
- `00-00-00-00-00-22-CA`
### 1.3 Revision History

<table>
<thead>
<tr>
<th>Revision</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>07/16/2012</td>
<td>Initial Release</td>
</tr>
<tr>
<td>2</td>
<td>03/18/2013</td>
<td>Numerous small changes</td>
</tr>
<tr>
<td>3</td>
<td>10/22/2013</td>
<td>Added exec reset system, exec sendResponse command; Other minor corrections</td>
</tr>
<tr>
<td>4</td>
<td>04/04/2014</td>
<td>Documented exec commands: setNumParents, failover, promoteToOperational;</td>
</tr>
<tr>
<td>5</td>
<td>10/28/2014</td>
<td>Clarified maxMotes description; Clarified decommissionDevice command; Added get commands: redundancy and sourceroute</td>
</tr>
<tr>
<td>6</td>
<td>04/22/2015</td>
<td>Updated get and set network descriptions; Updated blacklisting requirements</td>
</tr>
<tr>
<td>7</td>
<td>12/03/2015</td>
<td>Minor corrections</td>
</tr>
</tbody>
</table>
2 Introduction

This guide describes the commands used to communicate with the LTP5903CEN-WHR SmartMesh WirelessHART manager through its command line interface (CLI). The CLI is available by connecting a serial terminal program to the Manager. The CLI is intended for human interaction with a manager, e.g. during development, or for interactive troubleshooting. Most commands are atomic - a command and its arguments are typed into the CLI, and a response is returned. For example, the `help` command returns a list of possible commands. Traces are not atomic - once started, they generate output asynchronously until cancelled.

For a machine-to-machine communications (e.g. a host program talking to the manager), the SmartMesh WirelessHART Manager API Guide is used. See the API guide for details on that interface.

2.1 CLI Access

Two types of CLI commands can be sent to the manager. The commands are accessed through separate CLI utilities:

- **Manager CLI Commands** allow you to view information about the network and perform administrative tasks, such as upgrading software on the motes and configuring a mote. These commands are available through the manager CLI utility.
- **Radio Test CLI Commands** are used for testing the manager’s radio transmission and reception for certification purposes. These commands are available through the radio test utility.

You can log on to either CLI serially using a serial terminal program (such as HyperTerminal) or a secure shell (SSH) session via Ethernet. Refer to the user guide for the program you have chosen. The manager has several interfaces that provide access to the command line:

- **Serial 2**—The serial interface can be used to access a CLI using a serial terminal program. The **Serial 2** port is a standard 9-pin D-SUB RS232 port. For hardware specifications and connection details, refer to the SmartMesh WirelessHART User’s Guide or the product datasheet. The default serial port settings are as follows: 115200 baud, 8 data bits, No parity, 1 stop bit, no flow control.

- **10Base-T Ethernet**—Use this interface to log on to a CLI using SSH. The connection details will vary depending upon how you have connected the manager to your LAN.
3 Manager CLI Commands

This section describes how to use the CLI commands for the SmartMesh WirelessHART Manager.

3.1 Logging on to the Manager CLI

The manager CLI is a separate shell that is run from the Linux command line. The manager CLI supports a maximum of three concurrent sessions. Attempting to launch any additional sessions will result in an error message stating that the maximum number of sessions are currently running. The root user is always permitted to log in to the manager CLI, if the maximum number of sessions has already been reached, then whichever session is oldest will be logged out to allow for the root user to log in.

To log on to the manager CLI:

At the prompt within the terminal program or SSH session, access the Linux logon prompt by entering the following username and password:

Username: dust
Password: dust

This will bring you to the Linux prompt. At the Linux prompt ($):

1. Enter: nwconsole
2. Enter the manager CLI user name and password. The default user name and password is:
   Username: admin
   Password: admin

⚠️ The default username and password can be changed using the set command. For instructions, see Managing Users and Passwords.

Alternatively, you can access nwconsole directly from the terminal program or SSH session prompt in Step 1 by entering the following username and password:

Username: dustcli
Password: dustcli
To log out of the Manager CLI:

To log out of the manager CLI and return to the Linux prompt, enter the `logout` command. If the SSH connection is dropped, the manager CLI session is logged out.

### 3.1.1 CLI Timeouts

Manager CLI sessions are logged out if they have been inactive for too long. The CLI timeout is specified in the `System` element with the `cliTimeout` field. The default timeout value is 120 minutes. If `cliTimeout` is set to 0, CLI sessions will never timeout.
3.2 delete

Description
Use this command to delete a mote from the Access Control List (ACL) or the network, or delete a username from the access list for the manager XML API.

Syntax

```plaintext
delete { acl #<mac> | mote {<id> | #<mac>} | user <userName>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>acl #&lt;mac&gt;</td>
<td>Deletes a mote (identified by its MAC address) from the access control list. If the security mode is set to <code>acceptACL</code>, deleting a mote from the ACL prevents the mote from joining the network, or rejoining if the mote is already in the network; it does not force a mote to leave the network.</td>
</tr>
<tr>
<td>mote &lt;id&gt;</td>
<td>#&lt;mac&gt;</td>
</tr>
<tr>
<td>user &lt;username&gt;</td>
<td>Removes a user (identified by the username) from the access list for manager’s XML API. Deleting a user prohibits that user from accessing the manager XML API.</td>
</tr>
</tbody>
</table>

Example

To delete the mote with MAC address `00-00-00-00-00-00-62-4C` from the ACL, enter:

```
delete acl #00-00-00-00-00-00-62-4C
```

To delete the mote with MAC address `00-00-00-00-00-00-62-4C` from the network, enter:

```
delete mote #00-00-00-00-00-00-62-4C
```

To delete the username `YuriZ` from the list of users who may access manager, enter:

```
delete user YuriZ
```
### 3.3 deleteSessions

**Description**
Use this command to close all control and notification sessions established by a client (identified by the client IP address). To verify that the sessions have been deleted, use the `show sessions` and `show cli-sessions` commands.

**Syntax**

```
deleteSessions <ip>
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ip</td>
<td>The client IP address for which control and notification sessions are to be closed.</td>
</tr>
</tbody>
</table>

**Example**

```
3.4 exec

The exec command allows you to execute the API commands described in this section.

3.4.1 exec activateAdv

Description
Use this command to trigger manager to turn on advertising for all motes in the network.

Syntax

```
exec activateAdv <mac> <timeout>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mac</td>
<td>MAC address must be set to ff-ff-ff-ff-ff-ff-ff-ff (turns advertising on for all motes)</td>
</tr>
<tr>
<td>timeout</td>
<td>The number of minutes (from 0 to 255) that advertising should remain on. After this period, advertising is turned off. Specifying 0 turns off advertising.</td>
</tr>
</tbody>
</table>

Example
To activate advertising for a 60 minute advertising period, enter:

```
> exec activateAdv FF-FF-FF-FF-FF-FF-FF-FF 60
activateAdvertising command has been accepted.
```
3.4.2 exec activateFastPipe

Description
Use this command to activate a pipe between the manager and a mote. A pipe is a high-bandwidth frame that allows for faster data transfer between the manager/gateway and the mote. Only one pipe may be active in the network at a time. To see the pipe status for a specific mote, use the `get mote` command.

You may want to activate a pipe if you are planning a large file transfer to a mote, or will be performing a series of request and response maintenance commands to a single mote.

Syntax

```
exec activateFastPipe <mac> <pipeDirection>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mac</td>
<td>MAC address of mote for which pipe is being activated</td>
</tr>
<tr>
<td>pipeDirection</td>
<td>Sets the direction of the pipe:</td>
</tr>
<tr>
<td></td>
<td>0 = Activates an upstream pipe (mote to manager)</td>
</tr>
<tr>
<td></td>
<td>1 = Activates a downstream pipe (manager to mote)</td>
</tr>
<tr>
<td></td>
<td>2 = Activates a pipe in both upstream and downstream directions</td>
</tr>
</tbody>
</table>

Example
To activate a bidirectional pipe between manager and a mote with MAC address of 00-00-00-00-00-40-C9, enter:

```
> exec activateFastPipe 00-00-00-00-00-40-C9 2
activateFastPipe command has been accepted
```
3.4.3 exec cancelOtap

Description
Cancels an OTAP in progress.

Use this command to cancel a software update (OTAP). If the OTAP has already reached the commit stage, the new software is being activated on the motes and an error message informs you that the OTAP cannot be cancelled. Before issuing the exec cancelOtap command, you may want to view the status of the OTAP by issuing the otap status command.

Syntax

exec cancelOtap

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example
To cancel an OTAP, enter:

exec cancelOtap
3.4.4 exec deactivateFastPipe

Description
Use this command to deactivate a pipe between the manager and a mote. Only one pipe may be active in the network at a time. To see the pipe status for a specific mote, use the `get mote` command.

Deactivate a pipe when it is no longer needed in order to conserve energy.

Syntax

```
exec deactivateFastPipe <mac>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mac</td>
<td>MAC address of mote for which pipe is being deactivated</td>
</tr>
</tbody>
</table>

Example
To deactivate a pipe between manager and a mote with MAC address of 00-00-00-00-00-40-C9, enter:

```
> exec deactivateFastPipe 00-00-00-00-00-00-40-C9
deactivateFastPipe command has been accepted.
```
3.4.5 exec decommissionDevice

Description
The `exec decommissionDevice` command causes the manager to prepare a mote (identified by its MAC address) for safe removal from the network. The target mote’s children are moved to other parents, and the node is designated as non-routing (it does not forward any other mote’s traffic). This makes it safe to remove the target mote from the network without disrupting other network traffic. The mote enters the `disconnected` state, which can be used to inform a user that it is safe to power down the mote and physically remove it from the network. Note that the `decommissionDevice` command can be used on any mote unlike the `delete` command, which can only be used to remove an inactive mote.

Syntax
```
exec decommissionDevice <MAC>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>The MAC address of the device to be decommissioned</td>
</tr>
</tbody>
</table>

Example
To decommission the mote with MAC address `00-00-00-00-00-62-4C`, enter:
```
exec decommissionDevice 00-00-00-00-00-00-62-4C
```
3.4.6 exec exchangeJoinKey

Description

Use this command to initiate the replacement of the common join key for security reasons. The process may take several minutes to complete, depending on network size. The join key is used by motes when they first join the network or communicate with a new mote.

The `exchangeJoinKey` command is only valid if the network security mode is set to accept the common join key. If the security mode is set to accept the Access Control List (ACL), each mote is assigned a separate join key and the common join key is not used. To see the current security mode, use the `get security` command.

Syntax

```
exec exchangeJoinKey <key>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>key</td>
<td>The new 16-byte join key. Each byte is entered as a pair of hexadecimal digits</td>
</tr>
</tbody>
</table>

Example

To change the join key to `3987A553544E4B4F4B4A5152544E434F`, enter:

```
exec exchangeJoinKey 3987A553544E4B4F4B4A5152544E434F
```
3.4.7 exec exchangeMoteJoinKey

Description

Use the exec exchangeMoteJoinKey command to initiate the replacement of the join key for a mote on the Access Control List (ACL). Note that you must use the same join key that you have specified for the mote in the ACL. Depending on the network size, it may take several minutes to complete the process of exchanging the join key on the mote.

The exchangeMoteJoinKey command is only valid if the network security mode is set to accept an ACL. This mode allows individual motes to have their own join key, and only motes on the ACL will be accepted into the network. The join key is used by motes when they first join the network.

To see the current ACL, use the get acls command. To change the join key for a mote on the ACL, use the set acl joinKey command. To see the current security mode, use the get security command.

Syntax

exec exchangeMoteJoinKey <MAC> <key>

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>MAC address of mote to receive the new join key</td>
</tr>
<tr>
<td>key</td>
<td>The new 16-byte join key. Each byte is entered as a pair of hexadecimal digits</td>
</tr>
</tbody>
</table>

Example

To change the join key to 1987A553544E4B4F4B4A5152544E434F on a mote with MAC address 00-00-00-00-00-62-4C, enter:

exec exchangeMoteJoinKey 00-00-00-00-00-62-4C 1987A553544E4B4F4B4A5152544E434F
3.4.8 exec exchangeNetworkId

Description
This command initiates the process of changing the Network ID on the manager and all its network motes. The process may take several minutes to complete, depending on network size. Allow time for the exchange to be completed before restarting the network.

If you want to operate two or more networks within range of each other, each network must have a different Network ID in order to preserve network integrity. The manager only allows motes to join the network if they share its Network ID and join key. The new Network ID takes effect the next time the network is restarted.

Syntax

```
exec exchangeNetworkId <network ID>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>network ID</td>
<td>Replaces the current Network ID with the specified Network ID. The Network ID can be any number between 1 and 65534</td>
</tr>
</tbody>
</table>

Example
To change the Network ID to 1668, enter:

```
exec exchangeNetworkId 1668
```
### 3.4.9 `exec exchangeMoteNetworkId`

**Description**
This command initiates the process of changing the Network ID on a specified mote in the network. The new ID takes effect on next mote join.

**Syntax**
```
exec exchangeMoteNetworkId <mac> <network ID>
```

**Parameters**
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>network ID</td>
<td>Replaces the current Network ID with the specified Network ID. The Network ID can be any number between 1 and 65534</td>
</tr>
<tr>
<td>mac</td>
<td>MAC address of mote to receive the new Network ID</td>
</tr>
</tbody>
</table>

**Example**
```
exec exchangeMoteNetworkId 00-00-00-00-00-00-62-4C 100
```
3.4.10 exec exchangeNetworkKey

Description
Use this command to initiate the process of replacing the current network key with a new randomly generated key for security reasons. The process takes several minutes to complete. The network key is used for link layer authentication between devices.

Syntax

exec exchangeNetworkKey

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

exec exchangeNetworkKey
3.4.11 exec exchangeSessionKey

Description
Use this command to trigger manager to generate and distribute a new random session key for the session between the manager (or gateway) and a mote (or motes).

The session key is used to encrypt all messages exchanged between the manager (or gateway) and the mote(s).

Syntax
exec exchangeSessionKey <macA> <macB>

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>macA</td>
<td>Identifies the MAC address of the manager (or gateway) for which the session key is to be generated. For the manager MAC address use: 00-1B-1E-F9-80-00-00-01 For the gateway address use: 00-1B-1E-F9-81-00-00-02</td>
</tr>
<tr>
<td>macB</td>
<td>Identifies the mote for which the session key is to be generated. The broadcast session key is distributed if you set ff-ff-ff-ff-ff-ff-ff-ff as the mote address.</td>
</tr>
</tbody>
</table>

Example
To change the session key for the session between a manager with MAC address 00-1B-1E-F9-80-00-00-01 and a mote with MAC address 00-00-00-00-00-00-40-C9, enter:

exec exchangeSessionKey 00-1B-1E-F9-80-00-00-01 00-00-00-00-00-00-40-C9
3.4.12 exec failover

**Description**

The `exec failover` command initiates a redundancy failover. If the Manager is currently the master, it will become the slave, and vice versa. Added in Manager >= 4.1.

**Syntax**

```
exec failover
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

**Example**

```
> exec failover
```

**Description**

The `exec failover` command initiates a redundancy failover. If the Manager is currently the master, it will become the slave, and vice versa. Added in Manager >= 4.1.

**Syntax**

```
exec failover
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

**Example**
> exec failover
3.4.13 `exec getLatency`

**Description**
Use this command to return an estimate of the roundtrip latency to communicate with a specific mote.

**Syntax**

```
exec getLatency <mac>
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mac</td>
<td>MAC address of mote for which latency is being estimated</td>
</tr>
</tbody>
</table>

**Example**

```
> exec getLatency 00-17-0D-00-00-01-02-03
Upstream latency = 3.73 seconds
Downstream latency = 1.280 seconds
```
3.4.14 exec getLicense

**Description**
Use this command to retrieve the current license. The license is returned in the following format: 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00.

**Syntax**
```
exec getLicense
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

**Example**
```
> exec getLicense
license: 00-00-00-00-00-00-00-00-00-00-00-00-00-00-00-00
```
3.4.15 exec getTime

**Description**

Use this command to request the current time as Universal Time (UTC) and Absolute Slot Number (ASN). The return value for the UTC time is in <seconds>.<microseconds>, which is the number of seconds and microseconds since midnight January 1, 1970.

**Syntax**

```bash
exec getTime
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

**Example**

```
> exec getTime
Current time (UTC) is: 1355509123.456297
Absolute Slot Number (ASN): 129723302
```
3.4.16 exec promoteToOperational

Description

The `exec promoteToOperational` command is used to move a mote from Quarantine state (in which it cannot send data) to the Operational state (in which it can). Available in manager >= 4.1

Syntax

```
exec promoteToOperational <MAC>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>MAC address of mote to promote</td>
</tr>
</tbody>
</table>

Example

```
exec promoteToOperational 00-00-00-00-00-12-34-56
```
Example

```
exec promoteToOperational 00-00-00-00-00-12-34-56
```
3.4.17 exec reset

Description

The `exec reset` command is used to manually reset a mote or the network, or clear statistics or the event log.

- `exec reset network`—Resets the manager’s wireless connection. As a result, the entire network must reform.
- `exec reset system`—Resets the manager process (needed for certain persistent parameters to take effect) and its wireless connection. As a result, the entire network must reform.
- `exec reset stat`—Clears all statistics on the network motes and manager.
- `exec reset eventLog`—Clears all events from the network events log. Events include activities such as system connection, motes joining the network, creation or deletion of paths and links, and alarms.
- `exec reset mote`—Resetting a mote turns the mote hardware off and on again, temporarily removing the mote from the network and allowing it to rejoin again and re-establish links to other motes.

Syntax

```
exec reset <obj> [<moteId>|#<mac>]
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
| obj       | Type of object to be reset:  
  - `network` = Resets manager’s wireless connection  
  - `system` = Resets manager process and manager’s wireless connection  
  - `stat` = Clears all statistics on the network motes and manager  
  - `eventLog` = Clears all events from the network events log  
  - `mote` = Causes a specific mote to reboot and rejoin the network |
| moteId #mac | Mote ID or MAC address of object to be reset |

Example

To reset the network (without resetting the manager process), enter:

```
exec reset network
```

To reset mote ID 19, enter:
exec reset mote 19
3.4.18 exec sendRequest

Description
Sends a request packet to a mote (identified by its MAC address). The binary data is sent as a string in which each byte is represented by two ASCII digits (0-9 and A-F).

Syntax

```
exec sendRequest <mac> <app> <priority> [-r] [-g<grID>] [-s<m1>:<m2>...]] <bytes>...
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mac</td>
<td>ReqParamDescr</td>
</tr>
<tr>
<td>app</td>
<td>Specifies the type of packet to be sent: publish = Data packets sent on a regular, periodic basis event = Packets (such as alarms) that are triggered by an event maintenance = A series of request/response packets used to manage the device blockTransfer = A group of packets sent over a period of time</td>
</tr>
<tr>
<td>priority</td>
<td>Specifies where the packet should be inserted in manager’s queue and sets packet priority within the network: low = Use low priority medium = Use medium priority high = Use high priority</td>
</tr>
<tr>
<td>-r</td>
<td>Sends the specified series of bytes as a reliable packet. A reliable packet must be acknowledged by the mote. If this field is not specified, a best-effort packet is sent.</td>
</tr>
<tr>
<td>-g&lt;grID&gt;</td>
<td>Graph to use in sending the packet. If this field is not specified, the default graph is used.</td>
</tr>
<tr>
<td>[-s&lt;m1&gt;:&lt;m2&gt;...]</td>
<td>Source route to use in sending the packet. The source route can include up to eight motes (identified by mote IDs).</td>
</tr>
<tr>
<td>bytes</td>
<td>The series of bytes to be sent as a hex string (for example 4f1a0823b6).</td>
</tr>
</tbody>
</table>
Example
To send the hexadecimal string 4f1a0823b6 as a reliable, low priority, maintenance packet to a mote with MAC address 00-00-00-00-00-40-C9, enter:

```
exec sendRequest 00-00-00-00-00-40-C9 maintenance low -r 4f1a0823b6
```
### 3.4.19 exec sendResponse

#### Description

Sends a response packet to a mote (identified by its MAC address). The binary data is sent as a string in which each byte is represented by two ASCII digits (0-9 and A-F).

#### Syntax

```
exec sendResponse <mac> <app> <priority> <callbackId> [-r] <bytes>
```

#### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>mac</td>
<td>MAC address of mote to which response packet is to be sent</td>
</tr>
<tr>
<td>app</td>
<td>Specifies the application domain of packet to be sent:</td>
</tr>
<tr>
<td></td>
<td>publish - Data packets to be sent on a regular, periodic basis</td>
</tr>
<tr>
<td></td>
<td>event - Packets (such as alarms) that are triggered by an event</td>
</tr>
<tr>
<td></td>
<td>maintenance - A series of request/response packets used to manage the device</td>
</tr>
<tr>
<td></td>
<td>blockTransfer - A group of packets sent over a period of time</td>
</tr>
<tr>
<td>priority</td>
<td>Specifies where the packet should be inserted in manager’s queue and sets packet priority within the network. Valid values are:</td>
</tr>
<tr>
<td></td>
<td>low</td>
</tr>
<tr>
<td></td>
<td>medium</td>
</tr>
<tr>
<td></td>
<td>high</td>
</tr>
<tr>
<td>callbackId</td>
<td>The callback ID of the packet as given in the original request packet from the mote or a user generated one if the response is unsolicited (a publish)</td>
</tr>
<tr>
<td>-r</td>
<td>Sends the specified series of bytes as a reliable packet</td>
</tr>
<tr>
<td>bytes</td>
<td>The series of bytes to be sent, as a hex string</td>
</tr>
</tbody>
</table>

#### Example

```
exec sendResponse 00170D0000123456 maintenance high 1234 -r 0000FC121234
```
3.4.20 exec serviceGetNext

Description

Get the next service for mote. Given a valid service ID, command will return the next service ID in the list corresponding to the source and destination addresses. A service ID of 0 can be used to get the first valid service ID.

Syntax

```
serviceGetNext {<MACA>|gw} {<MACB>|gw} <srvId>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MACA</td>
<td>The MAC address of the source mote or GW for which the service is associated</td>
</tr>
<tr>
<td>MACB</td>
<td>The MAC address of the destination mote or GW for which the service is associated</td>
</tr>
<tr>
<td>srvId</td>
<td>The starting ID for the search, or 0 to start a new search.</td>
</tr>
</tbody>
</table>

Example

```
> serviceGetNext F981000000000002 00170D000001234 0
```
3.4.21 exec serviceRequest

Description

Creates a service between devices. The source of the service is given in the first MAC parameter, and the destination in the second. To establish a session to/from the Gateway, the gateway's well known address of 001B1EF981000002 is used.

Syntax

serviceRequest {<MACA>|gw} {<MACB>|gw} <srvId> <app> <period> [source|sink|intermittent]

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MACA</td>
<td>MAC address of the source mote or Gateway</td>
</tr>
<tr>
<td>MACB</td>
<td>MAC address of the destination mote or Gateway</td>
</tr>
<tr>
<td>srvID</td>
<td>Unique identifier for this service to the destination mote.</td>
</tr>
<tr>
<td>app</td>
<td>The application domain for the service, one of:</td>
</tr>
<tr>
<td></td>
<td>publish</td>
</tr>
<tr>
<td></td>
<td>maintenance</td>
</tr>
<tr>
<td></td>
<td>event</td>
</tr>
<tr>
<td></td>
<td>block transfer</td>
</tr>
<tr>
<td>period</td>
<td>Interval between packets, in ms</td>
</tr>
<tr>
<td>type</td>
<td>The type of service, one of:</td>
</tr>
<tr>
<td></td>
<td>source - the source address is generating packets</td>
</tr>
<tr>
<td></td>
<td>sink - the source address is sinking packets</td>
</tr>
<tr>
<td></td>
<td>intermittent - service period is to be used as a latency target for infrequent traffic such as process alarms</td>
</tr>
</tbody>
</table>

Example

> serviceRequest 001B1EF981000002 00170D0000001234 127 maintainance 5000
3.4.22 exec setLicense

Description
Use this command to enter the software license string. The license will take affect after a Manager reset. For settings that require a reset to take affect and depend upon a license, this means that two resets are required - one for the license change, and one for the settings change.

Syntax

exec setLicense <license>

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>license</td>
<td>Software license string as 13-byte (26 character) hexadecimal string</td>
</tr>
</tbody>
</table>

Example

exec setLicense 00000000000000000000000000
3.4.23 exec setNumParents

Description

The `exec setNumParents` command changes the target number of steady-state parents for a node from the default value of 2. More parents increases the likelihood that a mote will stay in the network when there is poor path stability, at the cost of additional power for sending keepalives to additional parents. Added in Manager >4.1.

Syntax

```
exec setNumParents <numParents>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numParents</td>
<td>The target number of parents for each node</td>
</tr>
</tbody>
</table>

Example

```
> exec setNumParents 3
```

Description

The `exec setNumParents` command changes the target number of steady-state parents for a node from the default value of 2. More parents increases the likelihood that a mote will stay in the network when there is poor path stability, at the cost of additional power for sending keepalives to additional parents. Added in Manager >4.1.

Syntax

```
exec setNumParents <numParents>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>
numParents | The target number of parents for each node

Example

```plaintext
> exec setNumParents 3
```
3.4.24 exec startOtap

Description
Initiates the automatic OTAP (over-the-air-programming) process to update mote software. You can specify the number of times the manager retries the OTAP if the previous attempt fails.

Use `exec startOtap` to update software on the manager and network motes. Before using this command, you must first use the manager’s Admin Toolset utility to copy the software update package to the manager. To use Admin Toolset for this task, refer to the “Software Upgrade” section in the Admin Toolset Guide.

⚠️ Do not use `exec startOtap` without assistance from a Dust Networks support engineer. If used incorrectly, these commands can cause problems with the network or render it inoperable.

Syntax

```
exec startOtap [-n <numRetries>]
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n numRetries</td>
<td>Number of times manager tries upgrading the software on the mote if the previous attempt fails.</td>
</tr>
</tbody>
</table>

Example
To start the OTAP process with 10 retries, enter:

```
exec startOtap -n 10
```
3.5 get

Use the get commands to display information about the network status and current configuration settings.

3.5.1 get acl

Description
Use the get acl command to find out whether a specified mote (identified by its MAC address) is on the Access Control List (ACL). Prints an error if the mote is not on the ACL.

Syntax

get acl #<MAC>

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>MAC address (EUI-64) of the mote</td>
</tr>
</tbody>
</table>

Example
To find out if the mote with MAC address 00-17-0d-00-00-01-02-03 is on the ACL, enter:

```
> get acl #00-17-0d-00-00-01-02-03
Error retrieving ACL device #00-17-0d-00-00-01-02-03: 7/.
```

If it is on the ACL the MAC address will be returned:

```
> get acl #00-17-0D-00-00-38-0C-AE
MAC: 00-17-0D-00-00-38-0C-AE
```
### 3.5.2 get acls

**Description**  
The `get acls` command lists all motes on the Access Control List (ACL).

**Syntax**
```
get acls
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

**Example**
```
> get acls

MAC: 00-17-0D-00-00-38-0C-AE
```
3.5.3 get alarms

Description
Use the `get alarms` command to show all alarms.

Syntax

```
get alarms
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

```
> get alarms
Couldn't find any alarms.
```
3.5.4 get blacklist

Description
Use the `get blacklist` command to show all blacklisted channels.

Syntax

```
get blacklist
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

```
> get blacklist
frequency: 2480
```
3.5.5 get mote

Description
The `get mote` command shows information for a mote (identified by Mote ID or MAC address).

Syntax

```
mote <id> | #<MAC>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Mote ID</td>
</tr>
<tr>
<td>MAC</td>
<td>Mote MAC address</td>
</tr>
</tbody>
</table>

Example
To show information for Mote ID 17, enter:
```
> get mote 17
moteId: 17
macAddr: 00-17-0D-00-00-19-2D-B2
name: ...
```

To show information for the mote whose MAC address is 00-17-0D-00-00-01-02-03, enter:
```
get mote #00-17-0d-00-00-01-02-03
moteId: 14
macAddr: 00-17-0d-00-00-01-02-03
name: ...
```

Additional Information

The `get mote` command displays the following mote information.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>moteld</strong></td>
<td>Unique identifier assigned to a mote when it joins the network. This identifier may change from join to join.</td>
</tr>
<tr>
<td>-----------</td>
<td>----------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>macAddr</strong></td>
<td>The mote MAC address.</td>
</tr>
<tr>
<td><strong>name</strong></td>
<td>User-defined mote name.</td>
</tr>
<tr>
<td><strong>powerSource</strong></td>
<td>Indicates the mote’s power source (battery, line/recharge/power scavenging).</td>
</tr>
<tr>
<td><strong>dischargeCurrent</strong></td>
<td>The discharge current, discharge time, and recovery time are collectively used to describe the current sourcing capabilities of the three possible types of power supply feeding the mote:</td>
</tr>
<tr>
<td><strong>dischargeTime</strong></td>
<td>The maximum time (in seconds) that the power supply can sustain the discharge current before experiencing a voltage drop</td>
</tr>
<tr>
<td><strong>recoveryTime</strong></td>
<td>The time (in seconds) required by the power supply to recover from a power drain (for example, recharge its capacitors).</td>
</tr>
<tr>
<td><strong>enableRouting</strong></td>
<td>Indicates whether the mote may be configured for routing data from other motes</td>
</tr>
<tr>
<td><strong>productName</strong></td>
<td>Product name</td>
</tr>
<tr>
<td><strong>hwModel</strong></td>
<td>Hardware model number</td>
</tr>
<tr>
<td><strong>hwRev</strong></td>
<td>Hardware revision number</td>
</tr>
<tr>
<td><strong>swRev</strong></td>
<td>Software revision number</td>
</tr>
<tr>
<td><strong>isAccessPoint</strong></td>
<td>Indicates that this mote is an access point. An access point mote bridges the wireless mesh network to a larger data network (for example, an Ethernet or RS-485 network) and may physically reside in a gateway.</td>
</tr>
<tr>
<td><strong>numJoins</strong></td>
<td>Indicates the total number of times that the mote has joined the network since the last time the manager reset.</td>
</tr>
<tr>
<td><strong>state</strong></td>
<td>Indicates the mote state:</td>
</tr>
<tr>
<td>Idle</td>
<td>Mote has not been part of the network since the manager started</td>
</tr>
<tr>
<td>Lost</td>
<td>Mote is not currently part of the network</td>
</tr>
<tr>
<td>Negotiating1</td>
<td>Mote is in the process of joining the network</td>
</tr>
<tr>
<td>Negotiating2</td>
<td>Mote is in the process of joining the network</td>
</tr>
<tr>
<td>Connected</td>
<td>Mote is connected to the network</td>
</tr>
<tr>
<td>Operational</td>
<td>Mote is operational</td>
</tr>
<tr>
<td>Disconnected</td>
<td>Mote may be physically removed from network without affecting network</td>
</tr>
<tr>
<td>reason</td>
<td>Reason for mote state:</td>
</tr>
<tr>
<td>---------------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>None</td>
</tr>
<tr>
<td></td>
<td>Maximum number of motes has been reached</td>
</tr>
<tr>
<td></td>
<td>Mote is unreachable</td>
</tr>
<tr>
<td></td>
<td>Mote is not connected</td>
</tr>
<tr>
<td></td>
<td>Mote has a configuration error</td>
</tr>
</tbody>
</table>

| `joinTime`                      | Indicates the last time that the mote joined the manager. If the mote has not joined since the manager     |
|                                 | started, the `joinTime` will be zero.                                                                        |

| `voltage`                       | A floating point value containing the reading from the last battery measurement                              |

| `numNeighbors`                  | The number of neighbors (potential and connected) that the mote has                                         |

| `allocatedPkPeriod`             | The currently allocated bandwidth (in msec/packet) from mote to gateway                                      |

| `allocatedPipePkPeriod`         | The currently allocated bandwidth (in msec/packet) for the pipe. The value 0x01 (action pending) means the   |
|                                 | manager is in the process of changing the state of the pipe from on to off (or vice-versa).                 |

| `pipeStatus`                   | Indicates the status of the pipe at the mote                                                                |

| `advertisingStatus`            | Indicates the status of advertising at the mote                                                             |

| `needNeighbor`                 | Indicates whether a mote has only one parent and (or) has insufficient links. A well formed network should  |
|                                 | have only one mote (the “single-parent” mote in the first hop) with this flag on. If the flag is on for    |
|                                 | other motes, it indicates that an additional mote needs to be added near them.                              |
3.5.6 get motes

Description
The `get motes` command shows information for all motes.

Syntax

```
get motes
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example
> get motes
moteId: 1
macAddr: 00-17-0D-00-00-19-2D-B2
name:
powerSource: Line
dischargeCurrent: 0
dischargeTime: 0
recoveryTime: 0
enableRouting: true
productName:
hwModel: 3
hwRev: 37
swRev: 3.0.2-0
isAccessPoint: true
numJoins: 1
state: Operational
reason:
joinTime: 1354211891321
voltage: 0.00
numNeighbors: 1
allocatedPkPeriod: 0
allocatedPipePkPeriod: 480
pipeStatus: off
advertisingStatus: off
needNeighbor: false
apRdntMode: master
goodNeighbors: 1
locationTag: not supported

moteId: 2
macAddr: 00-17-0D-00-00-10-1F-08
name:
powerSource: Line
...

**Additional Information**

The `get motes` command displays the following mote information.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>moteId</td>
<td>Unique identifier assigned to a mote when it joins the network. This identifier may change from join to join.</td>
</tr>
<tr>
<td>macAddr</td>
<td>The mote MAC address.</td>
</tr>
<tr>
<td>name</td>
<td>User-defined mote name.</td>
</tr>
<tr>
<td>powerSource</td>
<td>Indicates the mote's power source (battery, line/recharge/power scavenging).</td>
</tr>
<tr>
<td>variable</td>
<td>Description</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>dischargeCurrent</td>
<td>The discharge current in mA.</td>
</tr>
<tr>
<td>dischargeTime</td>
<td>The maximum time (in seconds) that the power supply can sustain the discharge current before experiencing a voltage drop.</td>
</tr>
<tr>
<td>recoveryTime</td>
<td>The time (in seconds) required by the power supply to recover from a power drain (for example, recharge its capacitors).</td>
</tr>
<tr>
<td>enableRouting</td>
<td>Indicates whether the mote may be configured for routing data from other motes.</td>
</tr>
<tr>
<td>productName</td>
<td>Product name.</td>
</tr>
<tr>
<td>hwModel</td>
<td>Hardware model number.</td>
</tr>
<tr>
<td>hwRev</td>
<td>Hardware revision number.</td>
</tr>
<tr>
<td>swRev</td>
<td>Software revision number.</td>
</tr>
<tr>
<td>isAccessPoint</td>
<td>Indicates that this mote is an access point. An access point mote bridges the wireless mesh network to a larger data network (for example, an Ethernet or RS-485 network) and may physically reside in a gateway.</td>
</tr>
<tr>
<td>numJoins</td>
<td>Indicates the total number of times that the mote has joined the network since the last time the manager reset.</td>
</tr>
<tr>
<td>state</td>
<td>Indicates the mote state:</td>
</tr>
<tr>
<td>Idle</td>
<td>Mote has not been part of the network since the manager started.</td>
</tr>
<tr>
<td>Lost</td>
<td>Mote is not currently part of the network.</td>
</tr>
<tr>
<td>Negotiating1</td>
<td>Mote is in the process of joining the network.</td>
</tr>
<tr>
<td>Negotiating2</td>
<td>Mote is in the process of joining the network.</td>
</tr>
<tr>
<td>Connected</td>
<td>Mote is connected to the network.</td>
</tr>
<tr>
<td>Operational</td>
<td>Mote is operational.</td>
</tr>
<tr>
<td>Disconnected</td>
<td>Mote may be physically removed from network without affecting network.</td>
</tr>
<tr>
<td>reason</td>
<td>Reason for mote state:</td>
</tr>
<tr>
<td>None</td>
<td>None.</td>
</tr>
<tr>
<td>Maximum number of motes has been reached</td>
<td>Maximum number of motes has been reached.</td>
</tr>
<tr>
<td>Mote is unreachable</td>
<td>Mote is unreachable.</td>
</tr>
<tr>
<td>Mote is not connected</td>
<td>Mote is not connected.</td>
</tr>
<tr>
<td>Mote has a configuration error</td>
<td>Mote has a configuration error.</td>
</tr>
<tr>
<td>joinTime</td>
<td>Indicates the last time that the mote joined the manager. If the mote has not joined since the manager started, the joinTime will be zero.</td>
</tr>
<tr>
<td>voltage</td>
<td>A floating point value containing the reading from the last battery measurement.</td>
</tr>
<tr>
<td>numNeighbors</td>
<td>The number of neighbors (potential and connected) that the mote has.</td>
</tr>
<tr>
<td>Parameter</td>
<td>Description</td>
</tr>
<tr>
<td>-------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>allocatedPkPeriod</td>
<td>The currently allocated bandwidth (in msec/packet) from mote to gateway</td>
</tr>
<tr>
<td>allocatedPipePkPeriod</td>
<td>The currently allocated bandwidth (in msec/packet) for the pipe. The value 0x01 (action pending) means the manager is in the process of changing the state of the pipe from on to off (or vice-versa).</td>
</tr>
<tr>
<td>pipeStatus</td>
<td>Indicates the status of the pipe at the mote</td>
</tr>
<tr>
<td>advertisingStatus</td>
<td>Indicates the status of advertising at the mote</td>
</tr>
<tr>
<td>needNeighbor</td>
<td>Indicates whether a mote has only one parent and (or) has insufficient links. A well formed network should have only one mote (the “single-parent” mote in the first hop) with this flag on. If the flag is on for other motes, it indicates that an additional mote needs to be added near them.</td>
</tr>
</tbody>
</table>
3.5.7 get network

Description
The `get network` command shows network information, such as network name and Network ID.

Syntax
```
get network
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example
> get network

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>netName:</td>
<td>Describes the specific network installation.</td>
</tr>
<tr>
<td>networkId:</td>
<td>The Network ID binds motes to their network. This value must be the same for all motes in the network and for the manager.</td>
</tr>
<tr>
<td>optimizationEnable:</td>
<td>Indicates whether algorithms are enabled on the manager to continuously optimize network links to improve network performance. Dust Networks strongly recommends that optimization remain on (enabled) at all times.</td>
</tr>
<tr>
<td>maintStartTime:</td>
<td>Start of maintenance period. Statistics will be discarded during this period.</td>
</tr>
<tr>
<td>maintEndTime:</td>
<td>End of maintenance period.</td>
</tr>
<tr>
<td>maxMotes:</td>
<td>Maximum number of motes (including APs) allowed in network.</td>
</tr>
<tr>
<td>numMotes:</td>
<td>The number of motes (excluding gateway motes) that are in the “Live” state.</td>
</tr>
</tbody>
</table>

Additional Information

The `get network` command displays the following information.
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accessPointPA</td>
<td>Describes the RF output power setting of a mote or manager that has a power amplifier (the power amplifier is either on or off).</td>
</tr>
<tr>
<td>ccaEnabled</td>
<td>The clear channel access (CCA) value indicates whether network motes (including the access point mote) listen on a channel before they transmit.</td>
</tr>
<tr>
<td>requestedBasePkPeriod</td>
<td>Defines the base bandwidth that the manager should allocate for each mote. Note that requested base packet period does not include bandwidth requested through mote service requests and cannot be used to allocate bandwidth for services.</td>
</tr>
<tr>
<td>minServicesPkPeriod</td>
<td>Defines the maximum bandwidth that a single mote may be allocated for total non-pipe user requested bandwidth. This limits service requests from motes.</td>
</tr>
<tr>
<td>minPipePkPeriod</td>
<td>Limits bandwidth on all pipes (manager-API requested pipes and pipes requested in service requests).</td>
</tr>
<tr>
<td>bandwidthProfile</td>
<td>Determines the frame size (number of timeslots) for upstream, downstream, and advertising traffic. There are three profiles available: P1 = Normal profile P2 = Low-power profile Manual = Manual profile. The manual profile is an advanced mode to be used only as directed by an application noe. The manual profile lets you use the next three bytes to manually set the number of timeslots for upstream, downstream, and advertising traffic.</td>
</tr>
<tr>
<td>manualUSFrameSize</td>
<td>The manual upstream frame size, manual downstream frame size, and manual advertising frame size message bytes manually indicate the frame size (number of timeslots) for upstream traffic, downstream traffic, and advertising. These parameters are only applicable if the bandwidth profile is set to “manual.”</td>
</tr>
<tr>
<td>netQueueSize</td>
<td>The number of messages in the manager’s network management queue.</td>
</tr>
<tr>
<td>userQueueSize</td>
<td>The number of messages in the manager’s user command queue.</td>
</tr>
<tr>
<td>locationMode</td>
<td>Deprecated - always off.</td>
</tr>
<tr>
<td>backboneEnabled</td>
<td>High speed backbone is on (note this will dramatically increase mote power consumption)</td>
</tr>
<tr>
<td>backboneSize</td>
<td>Backbone superframe size - one of 8, 16</td>
</tr>
<tr>
<td>numParents</td>
<td>Number of upstream parents. Defaults to 2. Can be increased in cases where path stability is known to be marginal.</td>
</tr>
<tr>
<td>fastAdvPeriod</td>
<td>Advertising period when the network is in ‘fast’ mode (i.e. at startup, or when motes go lost)</td>
</tr>
<tr>
<td>slowAdvPeriod</td>
<td>Advertising period when the network is in ‘slow’ mode (i.e. 1 hour after last mote join)</td>
</tr>
</tbody>
</table>
3.5.8 get otapfiles

Description
The `get otapfiles` command lists the files that are currently being updated via OTAP (over-the-air programming).

Syntax

```
get otapfiles
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

```
get otapfiles
```
3.5.9 get otapmotes

Description
The `get otapmotes` command lists the motes that are receiving an OTAP update, and the status for each mote.

Syntax
```
get otapmotes
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example
```
get otapmotes
```
3.5.10 get otapstatus

Description
The `get otapstatus` command provides a summary of the OTAP status.

Syntax
```
get otapstatus
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example
```
> get otapstatus
state:      Idle
```
3.5.11 get paths

Description
The `get paths` command shows information about all network paths.

Syntax
```
get paths
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example
```
> get paths
pathId: 65514
moteAMac: 00-17-0D-00-00-19-2D-B2
moteBMac: 00-17-0D-00-00-38-0C-AE
numLinks: 6
pathDirection: upstream
pathQuality: 82.90

pathId: 65514
moteAMac: 00-17-0D-00-00-38-0C-AE
moteBMac: 00-17-0D-00-00-19-2D-B2
numLinks: 2
pathDirection: downstream
pathQuality: 82.90
...
```
3.5.12 get path

Description

The `get path` command displays the path ID and number of links for a path (identified by the MAC addresses of the motes on either end of the path).

Syntax

```
path <MACA> <MACB>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MACA</td>
<td>MAC address of mote at origin of path</td>
</tr>
<tr>
<td>MACB</td>
<td>MAC address of mote at end of path</td>
</tr>
</tbody>
</table>

Example

To display information for the path between mote 00-17-0d-00-00-01-02-03 and mote 00-17-0D-00-00-10-2E-F6, enter:

```
> path 00-17-0d-00-00-01-02-03 00-17-0D-00-00-10-2E-F6
pathId:                      65564
moteAMac:                    00-17-0d-00-00-01-02-03
moteBMac:                    00-17-0D-00-00-10-2E-F6
numLinks:                    6
pathDirection:               downstream
pathQuality:                 85.80
```
3.5.13 get redundancy

Description

The `get redundancy` command returns redundancy status of the Manager and the status and version of the peer (slave). Available in Manager version >= 4.1.3

Syntax

```
get redundancy
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

```
> get redundancy
localMode: master
peerStatus: unknown
peerControllerSwRev: 3.0.0.2
```
3.5.14 get security

Description
The `get security` command shows security information.

Syntax
```
get security
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example
```
> get security
securityMode:      acceptCommonJoinKey
acceptHARTDevicesOnly: false
```
3.5.15 get sla

Description
The `get sla` command shows the active network Service Level Agreement (SLA) settings (minimum network reliability, maximum network latency, and minimum path stability levels).

Syntax

```
get sla
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

```
> get sla
minNetReliability: 98.00
maxNetLatency: 12500
minNetPathStability: 50.00
apRandtCoverageThreshold: 70.00
```
### 3.5.16 get sourceroute

**Description**

The `get sourceroute` command returns the current primary and secondary source routes to a mote. Available in Manager version >= 4.1.3

**Synatax**

```
get sourceroute #<MAC>
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC</td>
<td>MAC address of the target mote</td>
</tr>
</tbody>
</table>

**Example**

```
> get sourceroute #00-17-0D-00-00-3F-FC-FF

destMac: 00-17-0D-00-00-3F-FC-FF

primary route:
  1. macAddr: 00-17-0D-00-00-20-80-98
  2. macAddr: 00-17-0D-00-00-3F-FC-FF

secondary route:
  1. macAddr: 00-17-0D-00-00-20-80-98
  2. macAddr: 00-17-0D-00-00-10-17-F4
  3. macAddr: 00-17-0D-00-00-3F-FC-FF
```
3.5.17 get system

Description
The `get system` command shows information about the system, such as system name, location, and software/hardware revision.

Syntax

```
get system
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

```
> get system
systemName: awesomeNet
location: dust
swRev: 4.0.1.33-1
hwModel: PM2511
hwRev: 006
serialNumber: 00170d80105b
time: 1355451359091
startTime: 1354211881000
cliTimeout: 120
controllerSwRev: 4.0.2.51
```
3.5.18 get users

Description
The `get users` command shows the username and privilege information for each network user.

Syntax
```
get users
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example
```
> get users
userName:  admin
privilege:  user
```
3.6 help

Description
Displays online help about a specified CLI command or all commands.

Syntax

```
help [<command> | -a ]
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a</td>
<td>Displays help for all commands</td>
</tr>
</tbody>
</table>

Example

To display help about the `reset` command:

```
help reset
```
3.7 logout

Description
Closes the current CLI session.

Syntax

```
logout
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

```
logout
```
3.8 onechan

Description
The `onechan` command sets the network to run on a single specified channel. The network continues to run on one channel until `onechannel` is turned off. The `onechan` command may be used for radio frequency compliance testing. Note that this command resets the Access Point (AP) mote, causing the network to re-form.

Syntax

```
onechan {off | <channel-number> }
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>off</td>
<td>Stops the network from running on one channel. The network returns to using all non-blacklisted channels.</td>
</tr>
<tr>
<td>channel-number</td>
<td>The channel (0–15) on which the network will run</td>
</tr>
</tbody>
</table>

Example

To run the network on channel 11, enter:

```
onechan 11
```

To stop the network from running on one channel and return to using all non-blacklisted channels, enter:

```
onechan off
```
### 3.9 otap

The Over-The-Air-Programming (OTAP) commands allow you to upgrade software on the motes and the gateway.

#### 3.9.1 otap start

**Description**

The `otap start` command performs the same function as `exec startOtap`. Refer to `exec startOtap` for a detailed description.

**Syntax**

```
otap start [-n <numRetries>] [dir]
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-n &lt;numRetries&gt;</td>
<td>Number of times manager tries upgrading the software on the mote if the previous attempt fails</td>
</tr>
<tr>
<td>[dir]</td>
<td>The OTAP directory containing the software upgrade file. The default OTAP directory is <code>/root/otap</code>.</td>
</tr>
</tbody>
</table>

**Example**

```
otap start -n 10
```
3.9.2 otap status

Description
Use this command to display the status of an OTAP session in progress. The following status conditions may be found:

- *None* = The OTAP process has been cancelled
- *Initializing* = OTAP handshakes are being performed in preparation for uploading files to the motes
- *Upload* = OTAP files are being uploaded to the motes
- *Commit* = The upload is complete and motes are being reset in order to activate the new software; cancelling an OTAP at this stage will result in motes running different software versions
- *Completed* = OTAP is completed

Syntax
```
otap status
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example
```
otap status
```
3.9.3 otap stop

**Description**

Use this command to pause an OTAP upload. To restart the OTAP upload, reissue the `otap start` or `exec startOtap` command.

**Syntax**

```
otap stop
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

**Example**

```
otap stop
```
3.10 ping

Description
This command sends a packet to a mote (or all motes) requesting a response. The mote returns its voltage and temperature. You can choose to ping a mote via a specific route or graph. If no route or graph is specified, the manager pings the mote using the optimum route. The ping command can be used to verify that a mote is in the network.

Syntax

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-q</td>
<td>Show only the totals; do not show individual pings</td>
</tr>
<tr>
<td>-c &lt;cnt&gt;</td>
<td>Number of pings to send (there is no limit to the number you may send)</td>
</tr>
<tr>
<td>-i &lt;int&gt;</td>
<td>Interval (in seconds) between pings</td>
</tr>
<tr>
<td>-g &lt;grId&gt;</td>
<td>Graph to use for the pings</td>
</tr>
<tr>
<td>-r <a href="">mId:mId...</a></td>
<td>Source route to use; the source route can include up to eight motes (identified by Mote IDs)</td>
</tr>
<tr>
<td>motId</td>
<td>ID of the mote to ping</td>
</tr>
<tr>
<td>#&lt;MAC&gt;</td>
<td>MAC address of mote to ping</td>
</tr>
</tbody>
</table>

Example
To ping Mote ID 20 ten times using the route through motes 7, 13 and 4, enter:

```
ping -c 10 -g 1 -r 7:13:4 20
```

To ping Mote ID 7 three times, enter:

```
ping -c 3 7
```
The following is an example of a ping result:

```
> ping -c 3 7
> [18:28:50] Ping note 7: reply #2: 2.496s 1 hops [24.0C 3.378V]
> [18:28:53] Ping note 7: reply #3: 2.496s 1 hops [24.0C 3.378V]
> [18:28:53] Ping note 7: sent 3, rcvd 3, 0% lost. Ave.roundtrip: 2.321s hops: 1
```

<table>
<thead>
<tr>
<th>Timestamp</th>
<th>Number of seconds for round trip</th>
<th>Number of hops from the manager</th>
<th>Temperature</th>
<th>Voltage</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3.10.1 ping show

Description
Shows the status of a ping command that is currently executing. If there is no ping pending, it will return "not found."

Syntax

```plaintext
ping show { all | ( <moteId> | #<MAC> )... }
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>all</td>
<td>Shows status of ping for all motes</td>
</tr>
<tr>
<td>moteId</td>
<td>Shows status of ping for a mote, identified by its ID</td>
</tr>
<tr>
<td>#&lt;MAC&gt;</td>
<td>Shows status of ping for a mote, identified by its MAC</td>
</tr>
</tbody>
</table>

Example

To check on the status of a ping command that was sent to Mote ID 20, enter:

```plaintext
ping show 20
```
3.10.2 ping stop

Description
Use this command to stop pinging all motes or a specific mote (identified by Mote ID or MAC address). Command will return the number of successful ping responses received.

Syntax

```
ping stop { all | { <moteId> | #<MAC> }... }
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>all</td>
<td>Stop pinging all motes</td>
</tr>
<tr>
<td>moteId</td>
<td>ID of mote to stop pinging</td>
</tr>
<tr>
<td>#&lt;MAC&gt;</td>
<td>MAC address of the mote to stop pinging</td>
</tr>
</tbody>
</table>

Example

To stop pinging Mote ID 28, enter:

```
> ping stop 28
[18:13:54] Ping mote 28: sent 16, rcvd 15, 6% lost. Ave.roundtrip: 3.254s hops:1
```
3.11 set

Description
Use this command to configure the following objects:

- Mote
- Network
- System
- User (user access)
- Security (network security)
- Sla (network service level agreement)
- Blacklist (network channel blacklist)
- Acl (Access Control List)

3.11.1 set acl

Description
Use this command to configure the Access Control List (ACL).

Syntax

```
set acl {#<macAddr> <joinKey>=<val>}
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>macAddr</td>
<td>MAC address (EUI-64) of mote to be added to the ACL</td>
</tr>
<tr>
<td>joinKey</td>
<td>Join key for the mote being added to the ACL</td>
</tr>
</tbody>
</table>

Example

```
> set acl #00:17:00:00:01:02:03 joinKey=0123456789ABCDEF0123456789ABCDEF
macAddr: 00-17-00-00-01-02-03
```
3.11.2 set blacklist

Description
Sets the channel blacklist, replacing any existing channel blacklist.

⚠️ For compliant operation in North America, channel 16 (2480 MHz) must be blacklisted. Dust modules are certified as Frequency Hopping devices - if you operate any device in the network at above +10 dBm EIRP, you may not be able to use blacklisting in some jurisdictions.

Syntax

```
set blacklist {<parameter>=<value>...}
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>frequency</td>
<td>The following frequency values may be specified: 2405 - 2480 in 5 MHz increments</td>
</tr>
</tbody>
</table>

Example

For example, to blacklist channels 2410, 2415, and 2445, enter:

```
> set blacklist frequency=2410 frequency=2415 frequency=2445
```

```
frequency: 2410
frequency: 2415
frequency: 2445
```
3.11.3 set mote

Description
The `set mote` command allows updating user-editable parameters of the mote. The mote must be identified with either its Mote ID or MAC address.

Syntax
```
set mote <moteId | #macAddr> <parameter>=<value>...
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>User-defined mote name</td>
</tr>
<tr>
<td>enableRouting</td>
<td>Sets mote configurability for routing:</td>
</tr>
<tr>
<td></td>
<td><code>true</code> = Mote can be configured for routing</td>
</tr>
<tr>
<td></td>
<td><code>false</code> = Mote cannot be configured for routing</td>
</tr>
</tbody>
</table>

Example
To set the name and `enableRouting` flag using the Mote ID:
```
> set mote 11 name=My_Mote enableRouting=false
```

To set just the `enableRouting` flag using the MAC address:
> set mote #00-17-0d-00-00-01-02-03 enableRouting=true

<table>
<thead>
<tr>
<th>Key</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>moteId</td>
<td>11</td>
</tr>
<tr>
<td>macAddr</td>
<td>00-17-0d-00-00-01-02-03</td>
</tr>
<tr>
<td>name</td>
<td>My_Mote</td>
</tr>
<tr>
<td>powerSource</td>
<td>Line</td>
</tr>
<tr>
<td>dischargeCurrent</td>
<td>0</td>
</tr>
<tr>
<td>dischargeTime</td>
<td>0</td>
</tr>
<tr>
<td>recoveryTime</td>
<td>0</td>
</tr>
<tr>
<td>enableRouting</td>
<td>false</td>
</tr>
<tr>
<td>isAccessPoint</td>
<td>false</td>
</tr>
<tr>
<td>numJoins</td>
<td>0</td>
</tr>
<tr>
<td>state</td>
<td>Idle</td>
</tr>
<tr>
<td>reason</td>
<td></td>
</tr>
<tr>
<td>allocatedPkPeriod</td>
<td>0</td>
</tr>
<tr>
<td>allocatedPipePkPeriod</td>
<td>480</td>
</tr>
<tr>
<td>pipeStatus</td>
<td>off</td>
</tr>
<tr>
<td>advertisingStatus</td>
<td>off</td>
</tr>
<tr>
<td>needNeighbor</td>
<td>false</td>
</tr>
<tr>
<td>goodNeighbors</td>
<td>0</td>
</tr>
<tr>
<td>locationTag</td>
<td>not supported</td>
</tr>
</tbody>
</table>
### 3.11.4 set network

**Description**
The `set network` command allows updating user-editable parameters of the network.

**Syntax**

```
set network {<parameter>=<value>...}
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>networkId</td>
<td>Network ID - HART recommends that some network IDs be reserved for specific purposes - see HCF Spec 75, table 2. 0xFFFF is never a valid network ID.</td>
</tr>
<tr>
<td>netName</td>
<td>Network name</td>
</tr>
<tr>
<td>joinKey</td>
<td>Common join key; this key is used for all motes when the security mode is set to accept the common join key</td>
</tr>
</tbody>
</table>
| optimizationEnable | Enables or disables network optimization: 
  - `true` = Enables optimization 
  - `false` = Disables optimization |
| maintStartTime | Reserved |
| maintEndTime | Reserved |
| maxMotes    | Maximum number of motes (including APs) allowed in network |
| accessPointPA | Sets the power amplifier on a 2.4 GHz network mote or manager that has a power amplifier: 
  - `true` = Turns the power amplifier on 
  - `false` = Turns the power amplifier off |
| ccaEnabled  | Sets clear channel access assessment (CCA): 
  - `true` = Enables CCA 
  - `false` = Disables CCA |
<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bandwidthProfile</td>
<td>Chooses which predefined bandwidth profile is used:</td>
</tr>
<tr>
<td></td>
<td>P1 (default): Upstream superframe = 1024 slots, Downstream = 256 slots</td>
</tr>
<tr>
<td></td>
<td>P2: Upstream = 1024 slots, Downstream = 2048 slots (slow network build, lower power steady state)</td>
</tr>
<tr>
<td>requestedBasePkPeriod</td>
<td>Reserved</td>
</tr>
<tr>
<td>minServicesPkPeriod</td>
<td>Reserved</td>
</tr>
<tr>
<td>minPipePkPeriod</td>
<td>Reserved</td>
</tr>
<tr>
<td>manualUSFrameSize</td>
<td>Reserved</td>
</tr>
<tr>
<td>manualDSFrameSize</td>
<td>Reserved</td>
</tr>
<tr>
<td>manualAdvFrameSize</td>
<td>Reserved</td>
</tr>
<tr>
<td>netQueueSize</td>
<td>Reserved</td>
</tr>
<tr>
<td>userQueueSize</td>
<td>Reserved</td>
</tr>
<tr>
<td>locationMode</td>
<td>Deprecated - do not use</td>
</tr>
<tr>
<td>backboneEnabled</td>
<td>Turns high-speed backbone on/off:</td>
</tr>
<tr>
<td></td>
<td><em>true</em> = Enables backbone</td>
</tr>
<tr>
<td></td>
<td><em>false</em> = Disables backbone</td>
</tr>
<tr>
<td>backboneSize</td>
<td>Backbone superframe size - one of 8, 16</td>
</tr>
<tr>
<td>numParents</td>
<td>Defaults to 2. Range of 1-4.</td>
</tr>
<tr>
<td>fastAdvPeriod</td>
<td>Interval between advertisements in fast mode (ms)</td>
</tr>
<tr>
<td>slowAdvPeriod</td>
<td>Interval between advertisements in slow mode (ms)</td>
</tr>
</tbody>
</table>

**Example**

To set the maximum number of motes to 24 and enable network optimization, enter:
> set network maxMotes=24 optimizationEnable=true

netName: myNet
networkId: 303
optimizationEnable: true
maxMotes: 24
numMotes: 1
accessPointPA: true
ccaEnabled: false
requestedBasePkPeriod: 100000
minServicesPkPeriod: 400
minPipePkPeriod: 480
bandwidthProfile: P1
manualUSFrameSize: 1024
manualDSFrameSize: 256
manualAdvFrameSize: 128
netQueueSize: 0
userQueueSize: 0
locationMode: off
### 3.11.5 set security

**Description**

Use the `set security` command to set the network security parameters.

**Syntax**

```
set security <parameter>=<value>...
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>securityMode</td>
<td>Sets the network security mode:</td>
</tr>
<tr>
<td></td>
<td><code>acceptACL</code> = Allows only the motes on the access control list to join the network</td>
</tr>
<tr>
<td></td>
<td><code>acceptCommonJoinKey</code> = Allows any mote that shares manager’s network join key to join the network</td>
</tr>
<tr>
<td>commonJoinKey</td>
<td>Sets the common join key for the network</td>
</tr>
</tbody>
</table>

**Example**

To set the security mode to accept only motes that are on the ACL, enter:

```
> set security securityMode=acceptACL
```

```
securityMode: acceptACL
acceptHARTDevicesOnly: false
```
### 3.11.6 set sla

#### Description
Use the `set sla` command to configure the network Service Level Agreement (SLA).

#### Syntax

```
set sla <parameter>=<value>...
```

#### Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>minNetReliability</td>
<td>Sets the minimum allowable network reliability (as percent). The percentage value is the number of packets received at manager divided by the number of packets expected at manager.</td>
</tr>
<tr>
<td>maxNetLatency</td>
<td>Sets the minimum allowable average packet latency (in seconds).</td>
</tr>
<tr>
<td>minNetPathStability</td>
<td>Sets the minimum allowable path stability (as percent). This percentage value is the number of acknowledgments received divided by the number of acknowledgments expected.</td>
</tr>
</tbody>
</table>

#### Example
To set the SLA for minimum network reliability to 98%, enter:

```
> set sla minNetReliability=98
```

```
minNetReliability:  98.00
maxNetLatency:      12500
minNetPathStability: 50.00
apRdntCoverageThreshold: 70.00
```
3.11.7 set system

Description
The set system command allows updating user-editable parameters of the system.

Syntax

```
set system <parameter>=<value>...
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>systemName</td>
<td>Name of system</td>
</tr>
<tr>
<td>location</td>
<td>Location of system</td>
</tr>
<tr>
<td>ctrlTcpPort</td>
<td>TCP port for the local plaintext control channel</td>
</tr>
<tr>
<td>ctrlSslPort</td>
<td>TCP port for the secure control channel</td>
</tr>
<tr>
<td>dataTcpPort</td>
<td>TCP port for local plaintext data</td>
</tr>
<tr>
<td>dataSslPort</td>
<td>TCP port for secure data</td>
</tr>
<tr>
<td>discoveryUdpPort</td>
<td>UPD port for manager discovery</td>
</tr>
<tr>
<td>sslEnabled</td>
<td>Enables security for the control and data channels. The sslEnable setting takes effect when the manager restarts.</td>
</tr>
</tbody>
</table>

- **true** = The ctrlTcpPort and dataTcpPort are only accessible from the localhost, and stunnel is used to provide encrypted access to ctrlSslPort and dataSslPort
- **false** = The ctrlTcpPort and dataTcpPort provide plaintext access for any client

Example
> set system systemName-awesomeNet

<table>
<thead>
<tr>
<th>systemName</th>
<th>awesomeNet</th>
</tr>
</thead>
<tbody>
<tr>
<td>location</td>
<td>dust</td>
</tr>
<tr>
<td>swRev</td>
<td>4.0.1.33-1</td>
</tr>
<tr>
<td>hwModel</td>
<td>PM2511</td>
</tr>
<tr>
<td>hwRev</td>
<td>006</td>
</tr>
<tr>
<td>serialNumber</td>
<td>00170d80105b</td>
</tr>
<tr>
<td>time</td>
<td>1355450339191</td>
</tr>
<tr>
<td>startTime</td>
<td>1354211881000</td>
</tr>
<tr>
<td>cliTimeout</td>
<td>120</td>
</tr>
<tr>
<td>controllerSwRev</td>
<td>4.0.2.51</td>
</tr>
</tbody>
</table>
3.11.8 set user

Description
Use the set user command to configure network users. Note that the password returned is the password hash, not the actual password.

Syntax

```
set user <parameter>=<value>...
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>userName</td>
<td>Username</td>
</tr>
<tr>
<td>password</td>
<td>User password</td>
</tr>
</tbody>
</table>
| privilege | Controls user access to the network:
viewver = Provides user access to a subset of CLI commands displaying manager information
user = Provides user access to all CLI commands |

Example

```
> set user username=JasonZ password=artZ2ba privilege=user
```

```
user Name: JasonZ
password: pWx3Drxqdi89c
privilege: user
```
3.12 show

3.12.1 show cli-sessions

Description
Shows CLI user names and login times.

Syntax

```
show cli-sessions
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

```
> show cli-sessions
admin       user  12/13/12 17:38:19
```
3.12.2 show events

Description
Shows events.

Syntax

```
show events <last> <num>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;last&gt;</td>
<td>Specifies the most recent event you want to see. If &lt;last&gt; is 0, shows the latest &lt;num&gt; events.</td>
</tr>
<tr>
<td>&lt;num&gt;</td>
<td>Specifies the number of events that you want to see which preceded the &lt;last&gt; event.</td>
</tr>
</tbody>
</table>

Example

To see 10 events with 3599 as the most recent event shown, enter:

```
> show events 3599 10
```

To see the latest 10 events, enter:

```
> show events 0 10
```

    1439 AlarmClose 12/11/12 10:30:00.507
    1430 PathDeact 12/10/12 17:03:46.074  65564  00-17-0D-00-00-19-2D-B2
    1431 PathDelete 12/10/12 17:03:46.076  65564  00-17-0D-00-00-19-2D-B2
    1432 MoteJoin 12/10/12 17:03:46.077    28  00-17-0D-00-00-38-0C-AE
    1433 AlarmClose 12/10/12 17:03:46.077   moteDown High  00-17-0D-00-0
    1434 PathCreate 12/10/12 17:03:46.078  65564  00-17-0D-00-00-19-2D-B2
    1435 PathAct 12/10/12 17:03:47.639  65564  00-17-0D-00-00-19-2D-B2
    1436 MoteLive 12/10/12 17:04:17.323    28  00-17-0D-00-00-38-0C-AE
    1437 SysDisconn 12/10/12 19:38:22.225  cli          admin         0.00
    1438 AlarmOpen 12/11/12 10:15:00.503     slaStability Low
3.12.3 show frame

Description
Shows information for the specified superframe. If no superframe number is specified, shows all superframe information.

Syntax

```
show frame <frId>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>frId</td>
<td>Superframe identifier</td>
</tr>
</tbody>
</table>

Example
> show frame
FRAME: 0
timeslots: 1024 (0-1023)
channels: 12 (0-11)
FRAME: 1
timeslots: 256 (0-255)
channels: 12 (0-11)
FRAME: 2
timeslots: 256 (0-255)
channels: 1 (14-14)
type: location
FRAME: 4
timeslots: 16 (0-15)
channels: 1 (13-13)
type: input pipe
FRAME: 5
timeslots: 16 (0-15)
channels: 1 (13-13)
type: output pipe
FRAME: 6
timeslots: 128 (0-127)
channels: 1 (12-12)
type: advertisement
Advertisement timeout 20000 msec.
Pipe Status: Off
3.12.4 show ap

Description
Shows information for the access point.

Syntax

```
show ap [-a]
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a</td>
<td>Includes information about unused paths</td>
</tr>
</tbody>
</table>

Example

To show information about the gateway and all unused paths, enter:
> show ap -a
00-17-0D-00-00-19-2D-B2     1     Oper SW: 3.0.2-0 HW: 37
Location is not supported
Number free TS: 838
Upstream hops: 0, latency: 0.000, TTL: 127
SourceRoute: Dist(Des):  0.0(1) Prim: 1 Sec:
Power Source: line
Advertisement Period: 20.000
Bandwidth:
    Summary for AP:  0.0433
Links per second: 11.914062 (unlimited)
    Broadcast links
      0. 1. 0: rtdb
      0.609. 2: rjb
<- #28   Links 4/4/4 RSSI: -32 Q: 0.87
    Broadcast links
      1. 24.10:  tb
      1. 51.11:  tb
      1. 83. 1:  tjb #28
      1.113. 7:  tb #28
      1.139. 5:  tb
      1.167. 7:  tb
      1.195.11:  tb
      1.225.10:  tb
      1.251. 8:  tb
<- #28   Links 2/2/2 RSSI: -32 Q: 0.87
    Broadcast links
      6.  0.12:  tb
      6. 16.12:  tb
      6. 32.12:  tb
      6. 48.12:  tb
      6. 62.12:  rjb
      6. 64.12:  tb
      6. 80.12:  tb
      6. 90.12:  rjb
      6. 96.12:  tb
      6.112.12:  tb
3.12.5 show linksmap

Description
Shows the list of links for each superframe and the corresponding end points for each link.

Syntax

```
show linksmap
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

```
> show linksmap
--- LinkMap for Frame#0 (timeslots: 0-1024) ---
ts=1 0:3
  ts=242 2:2
  ts=505 2:2
  ts=609 2:1
  ts=757 2:2
  ts=932 9:1
  ts=1020 2:2
--- LinkMap for Frame#1 (timeslots: 0-256) ---
ts=24 10:1
  ts=51 11:1
  ts=83 1:2
  ts=112 9:1
  ts=113 7:2
  ts=139 5:1
  ts=167 7:1
  ts=195 11:1
  ts=225 10:1
  ts=251 8:1
--- LinkMap for Frame#2 (timeslots: 0-256) ---
ts=255
```

...
3.12.6 show mote

Description
Shows information for a mote (identified by Mote ID or MAC address).

Syntax

```bash
show mote {<moteId>|#<MAC>} [-l] [-a]
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>moteId</td>
<td>Mote ID</td>
</tr>
<tr>
<td>MAC</td>
<td>Mote MAC address</td>
</tr>
<tr>
<td>-l</td>
<td>Show links</td>
</tr>
<tr>
<td>-a</td>
<td>Includes information about unused motes</td>
</tr>
</tbody>
</table>

Example

To show information for all motes, enter:

```bash
show motes -a
```

A shortcut for this command is `sm -a`. For an example of how mote information displays, see the `sm` command.

To show information about a particular mote, enter:
> show mote 28
00-17-0D-00-00-38-0C-AE  28  9 Oper SW: 1.0.1-53 HW: 33 (*)

Location is supported
Upstream hops: 1, latency: 1.602, TTL: 127
SourceRoute: Dist(Des): 1.2(0) Prim: 1,28 Sec:
Power Source: battery
Advertisement Period: 20.000
Bandwidth:
  output planned / current : 0.1333 / 0.1333
  global service / delta   : 0.0433 / 0.0000
  local service goal / current : 0.0433 / 0.0433
  guaranteed for services / child: 0.0433 / 0.0000
  free                    : 0.0900
Planned BW per parent: #1 - 0.0433
Mote services BW: 0.0433
  Problem/Fix Motes 65535/65535 status 0 cur ID 0
  Default BW: 0.0100
  ID: 0 Time: 30000 BW: 0.0333
Neighbors: 1 (max 32). Links: 10 (max 200)
Links per second: 2.539062 (limit: 21.329132)
  Broadcast links
  0. 1. 0: rtdb
  0.932. 9: rjb
  -> #1 T Links 4/4/4 RSSI: -33 Q: 0.88
  Broadcast links
  1.112. 9: tjb
  -> #1 Links 2/2/2 RSSI: -33 Q: 0.88
  Broadcast links
  6. 94.12: tb

Description of fields:

- Line 1: MAC address of mote, mote ID, time since last packet, state, software version, hardware type
- Line 3: Upstream hops, average upstream latency, average upstream TTL count
- Line 4: Source route average distance, list of primary source route addresses, list of secondary source route addresses (if one is calculated)
- Lines 6-16: Information about planned versus assigned bandwidth, in links/s needed to meet service and base bandwidth for that mote. The "limit" value is the peak packets/s reported by the mote at join.
- Line 17: Total neighbors and links, with device specific maximums
- Line 19+: Information about links, organized per frame. Contains frame, slot, offset, and flags, where r=receive, t=transmit, d=discovery, b=broadcast, j=join
3.12.7 show motes

Description
Shows information for all motes.

Syntax

show motes [-a]

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a</td>
<td>Includes motes in unused state</td>
</tr>
</tbody>
</table>

Example

To show all motes, including motes in unused state, enter:

```
> show motes -a
Current time: 12/13/12 17:46:21 ASN: 123769104
Elapsed time: 14 days, 07:48:20

<table>
<thead>
<tr>
<th>MAC</th>
<th>MoteID</th>
<th>Age</th>
<th>Jn</th>
<th>UpTime</th>
<th>Fr</th>
<th>Nbrs</th>
<th>Links</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>00-17-0D-00-00-19-2D-B2</td>
<td>ap</td>
<td>1</td>
<td>1</td>
<td>14:07:48:10</td>
<td>6</td>
<td>1</td>
<td>25</td>
<td>Oper</td>
</tr>
<tr>
<td>00-17-0D-00-00-10-0A-4D</td>
<td>21</td>
<td>-1</td>
<td>0</td>
<td>00:00:00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Idle</td>
</tr>
<tr>
<td>00-17-0D-00-00-10-0A-51</td>
<td>27</td>
<td>-1</td>
<td>0</td>
<td>00:00:00</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Idle</td>
</tr>
<tr>
<td>00-17-0D-00-00-38-0C-AE</td>
<td>28</td>
<td>15</td>
<td>2</td>
<td>3:00:42:33</td>
<td>2</td>
<td>1</td>
<td>10</td>
<td>Oper</td>
</tr>
</tbody>
</table>
```
3.12.8 show msg

Description
Shows how many reliable messages the manager has transmitted and how many messages it re-sent. If the number of repeated commands is high, it can indicate a problem sending downstream. Repeated commands should be less than 10% of the total number of commands. More than 10% may indicate the network is not stable.

Syntax

```
show msg [-v]
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-v</td>
<td>Includes any messages currently contained in manager's transmit queue</td>
</tr>
</tbody>
</table>

Example
To show the number of reliable messages the manager has transmitted and re-sent, including messages contained in the transmit queue, enter:

```
> show msg -v
** Commands in topology (approximately): 0
** Prepared topology/user packets: 0
** All packets (prepared and requested): 0
** Number of free buffers (max): 8 (8)
** Output Queue Size: 0 Mngr: 0 VGW: 0 Waiting: 0 User: 0
** Downstream stats: Reliable : 30. Reliable Repeats: 1/2
** Downstream stats: Tx Data : 0. Data rate (msgs/sec): 0. Avrg: 0. Max: 0
** Upstream stats: Rcvd Total : 10789. Rcvd linkfeedback :24
** Upstream stats: Rcvd Data Q : 8791. Data rate (msgs/sec): 0. Avrg: 0. Max: 0
APD queue maximum : 1
Input queue. Current : 0 Total: 921 Max: 0 Avrg: 0
Input Data queue. Current: 0 Total: 8791 Max: 0 Avrg: 0 AbsMax: 0
Num Packets Input/Output: 0 / 0
** Waiting Queue:
** Manager Queue:
** Gateway Queue:
** User Queue:
```
3.12.9 show optimization

Description
Show current optimization state.

Syntax

```
show optimization
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

```
> show optimization
Auto-recalculation: On
```
3.12.10 show sessions

Description
Shows how many XML sessions are currently running.

Syntax
show sessions

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

> show sessions
Control session limits:
  max external sessions: 100
  max cli sessions: 4
  max serial sessions: 1
  max datalog sessions: 5
  max web_proxy sessions: 5
Control sessions:
  1. user=admin           user      0.0.0.0         last use=12/13/12 17:38:19
Notification sessions:
# 3.12.11 show stat

**Description**
Displays statistics for motes, paths and the network.

**Syntax**
Use the following syntax to show statistics for a given interval:

```
show stat { cur | short <idx> | day <idx> | life }
```

Use the following syntax to show statistics for a given mote, path or network:

```
show stat { mote <moteId> | path <moteAId> <moteBId> | net }
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cur</td>
<td>Shows statistics for the current 15-minute interval for all motes, paths, and network.</td>
</tr>
</tbody>
</table>
| short <idx> | Shows 15-minute interval statistics for motes, paths, and network.  

<idx> specifies the interval you wish to view. The most recently completed 15-minute interval is 0. The oldest available interval is 95 (there are 96 15-minute intervals in a 24-hour period).  

For example, to view statistics for the most recently completed 15-minute interval, enter:  

```
show stat short 0
```

| day <idx> | Shows 1-day interval statistics for all motes, paths, and the network.  

<idx> specifies the day you wish to view. The current day is 0. Yesterday is 1.  

For example, to view statistics for the day before yesterday, enter:  

```
show stat day 2
```

<table>
<thead>
<tr>
<th>life</th>
<th>Shows lifetime statistics for all motes, paths, and network.</th>
</tr>
</thead>
</table>
| mote <moteId> | Shows all statistics for a specified mote (identified by its Mote ID).  

For example, to view statistics for mote 19, enter:  

```
show stat mote 27
```
### Example

To display all statistics for Mote ID 20, enter:

```bash
> show stat mote 28
```

<table>
<thead>
<tr>
<th>LIFETIME</th>
<th>Index</th>
<th>Start time</th>
<th>Id</th>
<th>Rx</th>
<th>Lost</th>
<th>Tx</th>
<th>Rx</th>
<th>Fwd</th>
<th>Drop</th>
<th>Dup</th>
<th>Lntcy</th>
<th>Jn</th>
<th>Hop</th>
<th>avQ</th>
<th>mxQ</th>
<th>me</th>
<th>ne</th>
<th>Chg</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>9651</td>
<td>12/13/12 17:30:00</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>637</td>
<td>1</td>
<td>1.56</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>5</td>
<td>02501</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>2310</td>
<td>12/13/12 17:30:00</td>
<td>0</td>
<td>2310</td>
<td>0</td>
<td>243</td>
<td>1.63</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>02501</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>3168</td>
<td>12/13/12 17:30:00</td>
<td>0</td>
<td>3168</td>
<td>0</td>
<td>202</td>
<td>1.59</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>3</td>
<td>01901</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>3168</td>
<td>12/13/12 17:30:00</td>
<td>0</td>
<td>3168</td>
<td>0</td>
<td>160</td>
<td>1</td>
<td>1.5</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>01084</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>1005</td>
<td>12/13/12 17:30:00</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>32</td>
<td>1.52</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0274 24</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>0</td>
<td></td>
<td>0</td>
<td>33</td>
<td>0</td>
<td>2</td>
<td>1.42</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>02510</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>0</td>
<td></td>
<td>0</td>
<td>33</td>
<td>0</td>
<td>2</td>
<td>1.41</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>02501</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>0</td>
<td></td>
<td>0</td>
<td>33</td>
<td>0</td>
<td>3</td>
<td>1.6</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>02493</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>0</td>
<td></td>
<td>0</td>
<td>33</td>
<td>0</td>
<td>3</td>
<td>1.42</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>02485</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>0</td>
<td></td>
<td>0</td>
<td>33</td>
<td>0</td>
<td>4</td>
<td>1.48</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>02476</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SHORT</th>
<th>Index</th>
<th>Start time</th>
<th>Id</th>
<th>Rx</th>
<th>Lost</th>
<th>Tx</th>
<th>Rx</th>
<th>Fwd</th>
<th>Drop</th>
<th>Dup</th>
<th>Lntcy</th>
<th>Jn</th>
<th>Hop</th>
<th>avQ</th>
<th>mxQ</th>
<th>me</th>
<th>ne</th>
<th>Chg</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>33</td>
<td>12/13/12 16:15:00</td>
<td>0</td>
<td>33</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1.42</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>02510</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>33</td>
<td>12/13/12 16:15:00</td>
<td>0</td>
<td>33</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1.41</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>02501</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>33</td>
<td>12/13/12 16:15:00</td>
<td>0</td>
<td>33</td>
<td>0</td>
<td>3</td>
<td>1.6</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>02493</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>33</td>
<td>12/13/12 16:15:00</td>
<td>0</td>
<td>33</td>
<td>0</td>
<td>3</td>
<td>1.42</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>02485</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>33</td>
<td>12/13/12 16:15:00</td>
<td>0</td>
<td>33</td>
<td>0</td>
<td>4</td>
<td>1.48</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>02476</td>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

To display statistics for motes, paths and the network during the most recently completed 15-minute interval, enter:
> show stat short 0
It is now .................. 12/13/12 17:50:22.
This interval started at ... 12/13/12 17:30:00.
There are 96 valid 15 minute intervals stored.
NOTE: you may be expecting packets that are still in transit!

--------------------------------NETWORK STATS----------------------------------
<table>
<thead>
<tr>
<th>PkArr</th>
<th>PkLost</th>
<th>PkTx(Fail/ Mic/ Seq)</th>
<th>PkRx</th>
<th>Relia.</th>
<th>Latency</th>
<th>Stability</th>
</tr>
</thead>
<tbody>
<tr>
<td>33</td>
<td>0</td>
<td>37( 4/ 0/ 0)</td>
<td>39</td>
<td>100%</td>
<td>1.42s</td>
<td>89.19%</td>
</tr>
</tbody>
</table>

---------------------------------MOTE STATS-----------------------------------
<table>
<thead>
<tr>
<th>Id</th>
<th>Rx</th>
<th>Lost</th>
<th>Tx</th>
<th>Rx</th>
<th>Fwd</th>
<th>Drop</th>
<th>Dup</th>
<th>Ltncy</th>
<th>Jn</th>
<th>Hop</th>
<th>avQ</th>
<th>mxQ</th>
<th>me</th>
<th>ne</th>
<th>Chg</th>
<th>T</th>
</tr>
</thead>
<tbody>
<tr>
<td>28</td>
<td>33</td>
<td>0</td>
<td>33</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>2</td>
<td>1.42</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>02510</td>
<td>25</td>
</tr>
</tbody>
</table>

---------------------------------PATH STATS-----------------------------------
<table>
<thead>
<tr>
<th>MoteA</th>
<th>MoteB</th>
<th>ABPower</th>
<th>BAPower</th>
<th>ABTx(Fail)</th>
<th>ABRx</th>
<th>BATx(Fail)</th>
<th>BARx</th>
<th>Stab.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>28</td>
<td>-33</td>
<td>-30</td>
<td>0( 0)</td>
<td>3</td>
<td>37( 4)</td>
<td>36</td>
<td>89.19%</td>
</tr>
</tbody>
</table>

Additional Information

The following information may display in response to the manager CLI `show stat` command. To display larger numerical values in the fixed-width tables, the suffix 'k' is used to denote 1,000 and the suffix 'M' to denote 1,000,000. Despite the displayed values being rounded off in these cases, the Manager internally stores the exact counts.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ABPower</td>
<td>The average RSSI values for transmissions from mote A to mote B.</td>
</tr>
<tr>
<td>ABRx</td>
<td>Number of packets mote B received from mote A.</td>
</tr>
<tr>
<td>ABTx(Fail)</td>
<td>Number of packets transmitted by mote A to mote B, and the number of packets for which mote A failed to receive acknowledgement. Packets may fail for a number of reasons, including RF interference or bit errors.</td>
</tr>
<tr>
<td>avQ</td>
<td>Average number of packets in the mote’s queue waiting to be transmitted.</td>
</tr>
<tr>
<td>BAPower</td>
<td>The average RSSI values for transmissions from mote B to mote A.</td>
</tr>
<tr>
<td>BARx</td>
<td>Number of packets mote A received from mote B.</td>
</tr>
<tr>
<td>BATx(Fail)</td>
<td>Number of packets transmitted by mote B to mote A, and the number of packets for which mote B failed to receive acknowledgement. (Not currently implemented. Acknowledgements are not sent for downstream data transmission.)</td>
</tr>
<tr>
<td>Chg</td>
<td>Charge consumption, in millicoulombs.</td>
</tr>
<tr>
<td>DnLat</td>
<td>Estimated latency for packets sent from the Manager to the mote. This value is based entirely on the hop depth of the mote and not on direct empirical measurement.</td>
</tr>
<tr>
<td>Hop</td>
<td>Average number of hops for this mote’s data packets to arrive at manager.</td>
</tr>
<tr>
<td>Id</td>
<td>Mote ID number.</td>
</tr>
<tr>
<td>Jn</td>
<td>Number of times the mote joined since manager was last reset.</td>
</tr>
<tr>
<td>----</td>
<td>---------------------------------------------------------------</td>
</tr>
<tr>
<td>me</td>
<td>MAC-layer MIC (message integrity check) errors.</td>
</tr>
<tr>
<td>MoteA MoteB</td>
<td>Describes the path from mote A to mote B.</td>
</tr>
<tr>
<td>mxQ</td>
<td>Maximum number of packets in the mote's queue waiting to be transmitted.</td>
</tr>
<tr>
<td>ne</td>
<td>Network-layer MIC (message integrity check) errors.</td>
</tr>
<tr>
<td>PkDrp</td>
<td>Number of data packets dropped by the mote. Packets may be dropped because the mote's buffer was full and the mote was not able to generate a packet.</td>
</tr>
<tr>
<td>PkDup</td>
<td>Number of duplicate packets received. A duplicate packet is sent if no acknowledgement is received for any hop in the packet's journey to manager. When duplicate packets are high, it indicates problems with the mote-to-mote communication.</td>
</tr>
<tr>
<td>PkFwd</td>
<td>Number of data packets forwarded by the mote to a neighbor.</td>
</tr>
<tr>
<td>PkLst</td>
<td>Number of data packets that manager expected, but did not receive.</td>
</tr>
<tr>
<td>PkRx</td>
<td>Total number of packets received by network motes.</td>
</tr>
<tr>
<td>PkTerm</td>
<td>Number of data packets terminated by the mote.</td>
</tr>
<tr>
<td>PkTx(Fail)</td>
<td>Total number of packets transmitted by network motes, and total number of packets for which no acknowledgement was received.</td>
</tr>
<tr>
<td>Reliability</td>
<td>Mote or network reliability.</td>
</tr>
<tr>
<td>Stability</td>
<td>Path stability (for overall network or motes A and B). Path stability is the percentage of data packets that are successfully transmitted and received by the next mote on the path. If a transmitting mote does not receive an acknowledgement, it resends the packet.</td>
</tr>
<tr>
<td></td>
<td>Manager calculates path stability for an interval by dividing the number of packets successfully transmitted by the sum of the number of packets successfully transmitted and the number of packets that failed to transmit.</td>
</tr>
<tr>
<td></td>
<td>$\text{Path stability} = \frac{\text{Packets Successfully Transmitted}}{\text{Packets Successfully Transmitted} + \text{Packets Failed}} \times 100%$</td>
</tr>
<tr>
<td>T</td>
<td>Temperature in Celsius</td>
</tr>
<tr>
<td>UpLat</td>
<td>Upstream data latency. The average time (in seconds) for a data packet to travel from the mote to manager. Manager calculates the data latency for each packet by subtracting the time the packet was received at manager from the packet timestamp, which indicates when the packet was generated by the mote. Manager averages the data latency of all packets from a mote to provide the values for network statistics.</td>
</tr>
</tbody>
</table>

**Examples**
The following is an example of the statistics provided in response to the `show stat life` command. For field descriptions, refer to the table above.

```
> show stat life
It is now ................. 04/08/13 09:29:03.
This interval started at ... 03/20/13 13:49:09.
--------------------------------NETWORK STATS----------------------------------
PkArr  PkLost  PkTx(Fail/ Mic/ Seq)  PkRx  Relia.  Latency   Stability
 4709k       5   35M(5888k/ 0/ 0)   45M    100%   3.53s       83.3%
---------------------------------MOTE STATS------------------------------------
Id    Rx  Lost    Tx    Rx   Fwd  Drop   Dup UpLat DnLat Jn Hop avQ mxQ me ne Chg  T
 23   17k     2   15k   22k   23k     3  2486  5.07     0  1  1  0  8  0  0 50k 27
 24   18k     0   16k  9204   86k    11 2685  3.29     0  1  1  0  9  0  0 51k 24
 25   18k     1   16k   22k   67k    31 2332  3.89     0  1  1  0  9  0  0 51k 23
 26   17k     2   15k   22k   28k     3  2734  4.76     0  1  1  0  9  0  0 59k 24
---------------------------------PATH STATS------------------------------------
MoteA MoteB ABPower BAPower  ABTx(Fail)  ABRx  BATx(Fail)  BARx   Stab.
 1    23    -60     -60       0(   0)  8880   28k(8109)   23k  71.95%
 1    24    -65     -71       0(   0)  23k   110k( 42k)   75k  61.43%
 1    25    -70     -70       0(   0)  274  1856( 748)  479k   59.7%
 1    26    -65     -67       0(   0)  22k   92k( 33k)   66k  63.56%
```

The following example shows how statistics are provided in response to the `show stat net` command. `Lifetime` shows lifetime statistics for all motes and paths. `Day` shows statistics for each day, starting with the current day. `Short` shows statistics for each 15-minute interval (an interval begins on the quarter hour), starting with the most recently completed 15-minute interval. For column heading descriptions, refer to the table above.
> show stat net

LIFETIME

<table>
<thead>
<tr>
<th>PkArr</th>
<th>PkLost</th>
<th>PkTx(Fail/ Mic/ Seq)</th>
<th>PkRx</th>
<th>Relia.</th>
<th>Latency</th>
<th>Stability</th>
</tr>
</thead>
<tbody>
<tr>
<td>4709k</td>
<td>40</td>
<td>35M(5888k/ 0/ 0)</td>
<td>45M</td>
<td>100%</td>
<td>3.53s</td>
<td>83.3%</td>
</tr>
</tbody>
</table>

DAY

<table>
<thead>
<tr>
<th>PkArr</th>
<th>PkLost</th>
<th>PkTx(Fail/ Mic/ Seq)</th>
<th>PkRx</th>
<th>Relia.</th>
<th>Latency</th>
<th>Stability</th>
</tr>
</thead>
<tbody>
<tr>
<td>94k</td>
<td>0</td>
<td>171k(116k/ 0/ 0)</td>
<td>2270k</td>
<td>100%</td>
<td>3.08s</td>
<td>85.19%</td>
</tr>
<tr>
<td>250k</td>
<td>0</td>
<td>1814k(257k/ 0/ 0)</td>
<td>2395k</td>
<td>100%</td>
<td>3.01s</td>
<td>85.84%</td>
</tr>
<tr>
<td>251k</td>
<td>0</td>
<td>1835k(278k/ 0/ 0)</td>
<td>2370k</td>
<td>100%</td>
<td>3.07s</td>
<td>84.85%</td>
</tr>
<tr>
<td>250k</td>
<td>0</td>
<td>1879k(322k/ 0/ 0)</td>
<td>2366k</td>
<td>100%</td>
<td>3.23s</td>
<td>82.82%</td>
</tr>
<tr>
<td>250k</td>
<td>0</td>
<td>1892k(335k/ 0/ 0)</td>
<td>2407k</td>
<td>100%</td>
<td>3.4s</td>
<td>82.3%</td>
</tr>
<tr>
<td>250k</td>
<td>0</td>
<td>1880k(318k/ 0/ 0)</td>
<td>2380k</td>
<td>100%</td>
<td>3.48s</td>
<td>83.05%</td>
</tr>
</tbody>
</table>

SHORT

<table>
<thead>
<tr>
<th>PkArr</th>
<th>PkLost</th>
<th>PkTx(Fail/ Mic/ Seq)</th>
<th>PkRx</th>
<th>Relia.</th>
<th>Latency</th>
<th>Stability</th>
</tr>
</thead>
<tbody>
<tr>
<td>index 0 .... start time 04/08/13 09:00:00</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2598</td>
<td>0</td>
<td>19k(3320/ 0/ 0)</td>
<td>25k</td>
<td>100%</td>
<td>3.85s</td>
<td>83.15%</td>
</tr>
<tr>
<td>2637</td>
<td>0</td>
<td>18k(2830/ 0/ 0)</td>
<td>24k</td>
<td>100%</td>
<td>3.29s</td>
<td>85%</td>
</tr>
<tr>
<td>2607</td>
<td>0</td>
<td>19k(2826/ 0/ 0)</td>
<td>25k</td>
<td>100%</td>
<td>3.12s</td>
<td>85.27%</td>
</tr>
<tr>
<td>2611</td>
<td>0</td>
<td>18k(2925/ 0/ 0)</td>
<td>24k</td>
<td>100%</td>
<td>3.17s</td>
<td>84.57%</td>
</tr>
<tr>
<td>2606</td>
<td>0</td>
<td>19k(3343/ 0/ 0)</td>
<td>25k</td>
<td>100%</td>
<td>3.44s</td>
<td>82.82%</td>
</tr>
</tbody>
</table>

| index 5 .... start time 04/08/13 07:45:00 |
| 2644  | 0      | 19k(3011/ 0/ 0)     | 25k  | 100%   | 3.1s    | 84.49%    |
| 2601  | 0      | 19k(3275/ 0/ 0)     | 25k  | 100%   | 3.46s   | 83.31%    |
| 2613  | 0      | 20k(3428/ 0/ 0)     | 25k  | 100%   | 3.21s   | 82.92%    |
| 2599  | 0      | 20k(3522/ 0/ 0)     | 25k  | 100%   | 3.41s   | 82.5%     |
3.12.12 show status

Description
Shows information about manager, including the IP address of each Ethernet interface, the number of motes and their status (live, connected).

Syntax
```
show status
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example
> show status
Current time: 12/13/12 17:50:50 ASN: 123796005
Elapsed time: 14 days, 07:52:49
Manager: 4.0.2.51
Redundant State: Master w/o Slave
Netlayer Mode: Master
**** APD ***
Gateway: 3.0.2-0. Connected. Hardware reset is ON
IP (eth0): 10.70.48.124
TS time: 10 ms
Offsets: 15
Mode: Steady State
AP Channel: 2
Location is not enabled
Channel mode is Regular
Max motes: 251
in network: 2
  live: 2
  connected: 0
no lost motes: 2
# of motes: 27
# activating: 0
Hardware CTS: Ready
Remote log: Log-service is off
UTC drift: 0.100
Usr/Tplg cmd ratio: 1/3
3.12.13 show time

Description
Shows the current time or translates the given ASN time to UTC.

Syntax

```
show time [<asn>]
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>asn</td>
<td>Absolute Slot Number (must be an integer)</td>
</tr>
</tbody>
</table>

Example

```
> show time
Current time: 12/13/12 17:51:26 ASN: 123799632
Elapsed time: 14 days, 07:53:25

> show time 415
ASN: 415 Time: 11/29/12 09:58:14
```
### 3.12.14 show ver

**Description**

Shows manager build and motes by software and hardware version.

**Syntax**

```plaintext
show ver
```

**Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

**Example**

To show version information, enter:

```plaintext
> show ver
4.0.2 (build 51) on 11/06/2012 14:12
1.0.1-53
  00-17-0D-00-00-38-0C-AE  28  Oper SW: 1.0.1-53 HW: 33 Stack: 1.0.1-1
3.0.2-0
  00-17-0D-00-00-19-2D-B2  1  Oper SW: 3.0.2-0 HW: 37
```

**Additional Information**

The following is an example of the information provided in response to the `show ver` command.

<table>
<thead>
<tr>
<th>Mote MAC address</th>
<th>Mote ID</th>
<th>Software version</th>
<th>Hardware version</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1.1 (build 55) on 10/06/2008 17:08</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00-17-0D-00-00-10-2E-F6  1  Oper SW: 2.1.1-154 HW: 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00-17-0D-00-00-10-13-D0  2  Oper SW: 2.1.1-154 HW: 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00-17-0D-00-00-10-14-3B  3  Oper SW: 2.1.1-154 HW: 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>00-17-0D-00-00-10-0C-4F  4  Oper SW: 2.1.1-154 HW: 1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3.13 sm

Description
This command shows information about all motes. It performs the same function as the show motes command described earlier in this guide.

Syntax

```
sm [-a]
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a</td>
<td>Includes motes in unused state</td>
</tr>
</tbody>
</table>

Example

```
> sm
Current time: 12/13/12 17:54:30 ASN: 123817979
Elapsed time: 14 days, 07:56:29

+-----------------+---------+---------+--------+-------+-------+-------+--------+
<table>
<thead>
<tr>
<th>MAC</th>
<th>MoteID</th>
<th>Age</th>
<th>Jn</th>
<th>UpTime</th>
<th>Fr</th>
<th>Nbrs</th>
<th>Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>00-17-0D-00-00-19-2D-B2</td>
<td>ap</td>
<td>1</td>
<td>1</td>
<td>14-07:56:18</td>
<td>6</td>
<td>1</td>
<td>25 Oper</td>
</tr>
<tr>
<td>00-17-0D-00-00-38-0C-AE</td>
<td>28</td>
<td>25</td>
<td>2</td>
<td>3-00:50:42</td>
<td>2</td>
<td>1</td>
<td>10 Oper</td>
</tr>
</tbody>
</table>
```

The following screen shows how mote information displays.
```plaintext
> sm -a
Current time: 12/13/07 16:12:15 ASN: 4397935
Elapsed time: 0 days, 10:02:01

<table>
<thead>
<tr>
<th>MAC</th>
<th>MotecID</th>
<th>Age</th>
<th>On</th>
<th>UpTime</th>
<th>FrNbrs</th>
<th>Links</th>
<th>State</th>
</tr>
</thead>
<tbody>
<tr>
<td>00-1E-00-00-00-01</td>
<td>ap</td>
<td>1</td>
<td>1</td>
<td>0-12:11:19</td>
<td>5</td>
<td>19</td>
<td>Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-02</td>
<td>10</td>
<td>336</td>
<td>1</td>
<td>0-12:09:21</td>
<td>2</td>
<td>2</td>
<td>10 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-03</td>
<td>12</td>
<td>18</td>
<td>1</td>
<td>0-12:09:03</td>
<td>2</td>
<td>3</td>
<td>10 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-04</td>
<td>20</td>
<td>100</td>
<td>1</td>
<td>0-12:09:00</td>
<td>2</td>
<td>2</td>
<td>10 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-05</td>
<td>3</td>
<td>54</td>
<td>1</td>
<td>0-12:09:55</td>
<td>2</td>
<td>5</td>
<td>13 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-06</td>
<td>14</td>
<td>183</td>
<td>1</td>
<td>0-12:10:30</td>
<td>2</td>
<td>2</td>
<td>10 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-07</td>
<td>13</td>
<td>252</td>
<td>1</td>
<td>0-12:07:46</td>
<td>2</td>
<td>2</td>
<td>10 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-08</td>
<td>5</td>
<td>13</td>
<td>1</td>
<td>0-12:09:04</td>
<td>2</td>
<td>3</td>
<td>10 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-09</td>
<td>9</td>
<td>240</td>
<td>1</td>
<td>0-12:07:20</td>
<td>2</td>
<td>4</td>
<td>11 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-0A</td>
<td>9</td>
<td>72</td>
<td>1</td>
<td>0-12:10:02</td>
<td>2</td>
<td>3</td>
<td>10 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-0B</td>
<td>13</td>
<td>257</td>
<td>1</td>
<td>0-12:09:09</td>
<td>2</td>
<td>2</td>
<td>10 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-0C</td>
<td>5</td>
<td>341</td>
<td>1</td>
<td>0-12:08:47</td>
<td>2</td>
<td>3</td>
<td>12 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-0D</td>
<td>2</td>
<td>151</td>
<td>1</td>
<td>0-12:08:01</td>
<td>2</td>
<td>2</td>
<td>10 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-0E</td>
<td>15</td>
<td>311</td>
<td>1</td>
<td>0-12:08:35</td>
<td>2</td>
<td>2</td>
<td>10 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-0F</td>
<td>11</td>
<td>107</td>
<td>1</td>
<td>0-12:08:42</td>
<td>2</td>
<td>3</td>
<td>11 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-10</td>
<td>17</td>
<td>153</td>
<td>1</td>
<td>0-12:08:40</td>
<td>2</td>
<td>6</td>
<td>14 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-11</td>
<td>4</td>
<td>23</td>
<td>3</td>
<td>0-12:07:03</td>
<td>2</td>
<td>3</td>
<td>11 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-12</td>
<td>15</td>
<td>126</td>
<td>1</td>
<td>0-12:10:20</td>
<td>2</td>
<td>4</td>
<td>11 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-13</td>
<td>7</td>
<td>56</td>
<td>1</td>
<td>0-12:08:50</td>
<td>2</td>
<td>2</td>
<td>10 Oper</td>
</tr>
<tr>
<td>00-1E-00-00-00-14</td>
<td>19</td>
<td>12</td>
<td>1</td>
<td>0-12:09:36</td>
<td>2</td>
<td>6</td>
<td>14 Oper</td>
</tr>
</tbody>
</table>
```
3.14 su

Description

This is a reserved command used for debugging

Syntax

su <string>

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>string</td>
<td>Reserved string</td>
</tr>
</tbody>
</table>

Example
3.15 trace

Description
Use this command to display subsystem information as it is generated. The information continues to display until you turn the trace off. To turn a trace off, type the command even if information is streaming in so quickly that you cannot see the command prompt.

The commands `trace io`, `trace rawio`, `trace ioall`, and `tracenotif` consume substantial resources on the manager and can cause performance degradation and network reset if they are left on. These commands should be used only for debugging purposes and turned off as soon as possible.

Syntax

```
trace <parameter> { on | off }
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>alarm</td>
<td>Displays information about active alarms</td>
</tr>
<tr>
<td>cong</td>
<td>Displays information about downstream congestion</td>
</tr>
<tr>
<td>io</td>
<td>Displays compressed input-output packets (a short version of rawio)</td>
</tr>
<tr>
<td>apmio</td>
<td>Low level Access Point Mote (APM) traffic</td>
</tr>
<tr>
<td>apmnak</td>
<td>APM NACK trace (Available Manager ≥ 4.1)</td>
</tr>
<tr>
<td>latency</td>
<td>Displays data latency</td>
</tr>
<tr>
<td>login</td>
<td>Displays the IP address of users that are currently logged on to the CLI</td>
</tr>
<tr>
<td>logmsg</td>
<td>Displays log messages</td>
</tr>
<tr>
<td>motest</td>
<td>Displays mote state changes</td>
</tr>
<tr>
<td>notif</td>
<td>Displays notification connection information</td>
</tr>
<tr>
<td>rawio</td>
<td>Displays raw input-output packets</td>
</tr>
<tr>
<td>rpc</td>
<td>Displays information about the xmlrpc connection</td>
</tr>
<tr>
<td>secio</td>
<td>Raw security I/O data</td>
</tr>
<tr>
<td>Command</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>stats</td>
<td>Displays statistics calculations</td>
</tr>
<tr>
<td>opt</td>
<td>Optimization scoring calculations (Available Manager &gt;= 4.1)</td>
</tr>
<tr>
<td>syserr</td>
<td>Displays system errors</td>
</tr>
<tr>
<td>topology</td>
<td>Displays topology events</td>
</tr>
<tr>
<td>thread</td>
<td>Displays thread start/stop events</td>
</tr>
<tr>
<td>timeout</td>
<td>Displays latency and average latency for communications with mote</td>
</tr>
<tr>
<td>c2cio</td>
<td>Controller-to-controller, i.e. redundancy (c2c) driver IO (Available Manager &gt;= 4.1)</td>
</tr>
<tr>
<td>c2crawio</td>
<td>Controller-to-controller driver raw IO (Available Manager &gt;= 4.1)</td>
</tr>
<tr>
<td>redun</td>
<td>Redundancy state (Available Manager &gt;= 4.1)</td>
</tr>
<tr>
<td>ioall</td>
<td>Displays all input and output information</td>
</tr>
<tr>
<td>info</td>
<td>Display standard information</td>
</tr>
<tr>
<td>bw</td>
<td>Bandwidth calculation (Available Manager &gt;= 4.1)</td>
</tr>
<tr>
<td>path</td>
<td>Path deletion (Available Manager &gt;= 4.1)</td>
</tr>
<tr>
<td>loc</td>
<td>Reserved (Deprecated in Manager &gt;= 4.1)</td>
</tr>
<tr>
<td>all</td>
<td>Displays information on all of the trace parameters</td>
</tr>
<tr>
<td>status</td>
<td>Shows which traces are currently running</td>
</tr>
</tbody>
</table>

**Example**

To turn on a trace for input and output packets, enter:

```
trace io on
```

When you are ready to turn this trace off, enter:

```
trace io off
```

To display raw input and output packets, enter:

```
trace rawio on
```
When you are ready to turn this trace off, enter:

```
trace rawio off
```

To display all input and output packets, enter:

```
trace all on
```

When you are ready to turn this trace off, enter:

```
trace all off
```
4 Access Point CLI Commands

This chapter describes how to use the Access Point CLI commands. These commands are used to test the manager’s radio transmission and reception. Use the following procedure to log on to the Access Point CLI.

4.1 Logging on to the Access Point CLI

Logging on to the Access Point CLI provides access to the radio test commands.

To log on to the Access Point CLI:

1. At the prompt within the terminal program or SSH session, access the Linux logon prompt by entering the username and password (the default is shown here).

   **Username:** dust  
   **Password:** dust

2. At the Linux prompt ($), enter:

   `/opt/dust-manager/bin/apdconsole.py`

3. You will be presented with a console message and a prompt (>):

   ```
   Conect to 127.0.0.1:55551
   >
   ```

To log out of the Access Point CLI and return to Linux:

1. Enter the `logout` command.
2. The CLI will print a message and exit:

   ```
   apdconsole exit!
dust@manager:~$ 
   ```
4.2 radiotest start

Description
Use this command to disable normal operation of manager’s access point mote in order to initiate radio testing using the `radiotest tx` and `radiotest rx` commands. Once in radiotest mode, only power cycling the manager or issuing `radiotest stop` will return the AP to normal operation.

Syntax

```
radiotest start
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

```
radiotest start
```
4.3 radiotest stop

Description
Use this command to return manager’s access point mote to normal operation after radio testing is completed.

Syntax

radiotest stop

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

radiotest stop
4.4 radiotest tx

Description

Use this command to initiate a series of 128-byte packet transmissions. As shown in the figure below, byte 0 contains the packet length (excluding the length parameter itself), which is 127 bytes. Bytes 1 and 2 contain the transmission sequence number. Bytes 3-125 contain a big-endian counter (from 0-122) that increments with every byte. The interpacket delay is 20 ms (see figure). Before using the radiotest tx command you need to issue the radiotest start command to place the manager’s Access Point (AP) mote in test mode. After radio testing is completed, issue the radiotest stop command to return manager’s access point mote to normal operation. If numPkts is set to 0, the AP generates an unmodulated test tone on the selected channel. The AP stops the tone with the radiotest stop command.

![Packet structure diagram]

**radiotest tx Packet Format**

⚠️ Channel numbering for this command is 0-15, corresponding to IEEE 2.4 GHz channels 11-26

Syntax

```bash
radiotest tx <channel> <numPkts>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel</td>
<td>RF channel (0 to 15) to use for the transmission test</td>
</tr>
<tr>
<td>numPkts</td>
<td>Number of packets to send. Specifying 0x00 sends an unmodulated test tone</td>
</tr>
</tbody>
</table>

Example
> radiotest tx 15 100

> TestRadioTx is responded with RC=0
4.5 radiotest set txpower

Description
The **radiotest set txpower** command sets the conducted transmit power level on the Access Point.

Syntax
```
radiotest set txpower <txPowerLevel>
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>txPowerLevel</td>
<td>Sets Tx power level. Valid values are -2 or 8.</td>
</tr>
</tbody>
</table>

Example
To turn power amplification on, enter:
```
> radiotest set txpower 8
> Set txPower RC=0
```
4.6 radiotest rx

Description

The radiotest rx command clears all previously collected statistics and initiates a test of radio reception for the specified channel and duration. During the test, the mote keeps statistics of the number of packets received (with and without error). The test results may be retrieved using the radiotest getstats command.

Before using the radiotest rx command you need to issue the radiotest start command to place manager’s access point mote in test mode. After radio testing and radio statistics retrieval is completed, issue the radiotest stop command to return manager’s access point mote to normal operation.

⚠️ Channel numbering for this command is 0-15, corresponding to IEEE 2.4 GHz channels 11-26

Syntax

radiotest rx <channel> <numSecs>

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>channel</td>
<td>RF channel (0 to 15) to use for the radio reception test</td>
</tr>
<tr>
<td>numSec</td>
<td>Duration or the radio reception test (in seconds)</td>
</tr>
</tbody>
</table>

Example

> radiotest rx 15 2
> TestRadioRx is responded with RC=0
4.7 radiotest getstats

Description

The `radiotest getstats` command retrieves statistics for the latest radio reception test performed using the `radiotest rx` command. The statistics show the number of good and bad packets (CRC failures) received during the test.

After radio testing is completed, you need to issue the `radiotest stop` command to return manager’s Access Point Mote to normal operation.

Syntax

```
radiotest getstats
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

```
> radiotest getstats
> rxTestStats: RC=0  rxOk=99  rxFail=1
```
4.8 show stats

Description
The `show stats` command displays statistics for the Access Point Driver (APD) and the Access Point Mote (APM).

Syntax

```
show stats
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example
> show stats
> *********** apd statistics ***********
  numDataTx      : 84
  numDataRetry   : 0
  numAckTx       : 15
  numNakTx       : 0
  numDataRx      : 15
  numDupDataRx   : 0
  numUnexpectedDataRx: 0
  numAckRx       : 84
  numNakRx       : 0
  maxAPMOutQSize : 1
  averAPMOutQSize: 0.133333
  averResponse   : 2.42857 ms
  averAPMToDCCDelay : 2.67857 ms
  numAPMToDCCPkt : 84

*********** apm statistics ***********
  numDataTx      : 13
  numDataRetry   : 0
  numAckTx       : 85
  numNakTx       : 0
  numDataRx      : 85
  numDupDataRx   : 0
  numAckRx       : 13
  numNakRx       : 0
  maxAPDOutQSize : 0
  averAPDOutQSize: 0
  averResponse   : 19 ms
  averDCCtoAPMDelay : 25.8333 ms
  numDCCtoAPMPkt : 12

Additional Information

The following information may appear in response to the radio test CLI show stats command.

<table>
<thead>
<tr>
<th>Field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>numDataTx</td>
<td>Number of packets transmitted</td>
</tr>
<tr>
<td>numDataRetry</td>
<td>Number of packets retransmitted</td>
</tr>
<tr>
<td>numAckTx</td>
<td>Number of acknowledgements transmitted</td>
</tr>
<tr>
<td>numNacTx</td>
<td>Number of packets for which no acknowledgement was transmitted</td>
</tr>
<tr>
<td>numDataRx</td>
<td>Number of packets receive</td>
</tr>
<tr>
<td>numDupDataRx</td>
<td>Number of duplicate packets received</td>
</tr>
<tr>
<td>NumUnexpectedDataRx</td>
<td>Number of unexpected packets received</td>
</tr>
<tr>
<td><strong>numAckRx</strong></td>
<td>Number of acknowledgements received.</td>
</tr>
<tr>
<td>-------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td><strong>numNacRx</strong></td>
<td>Number of packets for which no acknowledgement was received</td>
</tr>
<tr>
<td><strong>maxAPMOutQSize</strong></td>
<td>Displays internal counters used by Dust Networks application engineering for system troubleshooting</td>
</tr>
<tr>
<td><strong>maxAPDOutQSize</strong></td>
<td>Displays internal counters used by Dust Networks application engineering for system troubleshooting</td>
</tr>
<tr>
<td><strong>averAPMOutQSize</strong></td>
<td>Displays internal counters used by Dust Networks application engineering for system troubleshooting</td>
</tr>
<tr>
<td><strong>averAPDOutQSize</strong></td>
<td>Displays internal counters used by Dust Networks application engineering for system troubleshooting</td>
</tr>
<tr>
<td><strong>averResponse</strong></td>
<td>Average response time (milliseconds)</td>
</tr>
<tr>
<td><strong>averDCCToAPMDelay</strong></td>
<td>Average delay from the manager to the APM</td>
</tr>
<tr>
<td><strong>averAPMToDCCDelay</strong></td>
<td>Average delay from the APM to the manager</td>
</tr>
<tr>
<td><strong>numAPMToDCCPkt</strong></td>
<td>Number of packets sent from the APM to the manager</td>
</tr>
<tr>
<td><strong>numDCCToAPMPkt</strong></td>
<td>Number of packets sent from the manager to the APM</td>
</tr>
</tbody>
</table>
4.9 show status

Description
The `show status` command returns the current state of the Access Point Mote (APM) - either radio test or normal, and whether the manager has received the AP’s boot message.

Syntax

```
show status
```

Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
</tr>
</thead>
</table>

Example

```
> show status
apm mode: radio test
apm booted
```
5 Managing Users and Passwords

This section provides instructions for managing users and passwords for the SmartMesh WirelessHART Manager CLI and Linux console.

5.1 Managing Users and Passwords for the Manager CLI

You can create new users for the nwconsole application (Manager CLI). Two password options are available, which determine command access privileges for the CLI session. To create new users or change passwords or access privileges, you must log on with user privileges. Instructions are provided below.

<table>
<thead>
<tr>
<th>Access Privileges</th>
<th>Default Password</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>user</td>
<td>admin</td>
<td>Provides user access to all CLI commands.</td>
</tr>
<tr>
<td>viewer</td>
<td>viewer</td>
<td>Provides access to a subset of CLI commands displaying network information.</td>
</tr>
</tbody>
</table>

To create a new username and password:

1. Log on to the manager CLI with user privileges. For log on instructions, see Logging on to the Manager CLI.
2. To create a new username and password, enter:

   ```
   set user userName=<new user name> password=<new password> privilege=<user/viewer>
   ```

To change a user password or access privileges:

1. Log on to the Manager CLI with user privileges. For log on instructions, see Logging on to the Manager CLI.
2. To change the password for an existing user, enter:

   ```
   set user userName=<user name> password=<new password> privilege=<user/viewer>
   ```
5.2 Changing the Linux Login Password

You can change the default Linux login password (for the dust user). The new password must be 5 to 8 characters long, and should include a combination of upper and lower case letters and numbers.

To change the Linux login password:

1. At the prompt within the terminal program or SSH session, access the Linux login prompt by entering the current username and password (the default is shown here).
   
   **Username:** dust
   
   **Password:** dust

2. At the Linux prompt ( $ ), enter:

   ```
   passwd
   ```

3. When prompted, enter the old password.
4. Enter the new password.
5. Re-enter the new password.
6 Linux Console Commands

The manager software runs on a Linux platform. There are a number of Linux console commands supplied along with the manager software for system administration of the device.

6.1 Logging into the Linux console

At the prompt within the terminal program or SSH session, access the Linux login prompt by entering the following username and password:

*Username:* dust  
*Password:* dust

6.2 clear-motes

usage: clear-motes [--use-defaults]

The clear-motes command stops the Manager and removes persistent mote configuration, including mote names and mote ACL entries. By default, the common join key is preserved. If the `--use-defaults` option is passed to the scripts, it also removes the common join key.
6.3 config-login

usage: config-login [-p <port>] { enable | disable | status }

The `config-login` command is used to enable or disable the console (Linux command line) login on Serial 1. When console login is enabled, the Serial 1 port provides a standard login prompt.

The dust user can invoke the `config-login` command using `sudo`.

```
dust@manager$ sudo config-login disable
```
### 6.4 config-ppp

```
usage: config-ppp [-p <port>] [-l <local IP>] [-r <remote IP>] [-w] [ enable | enable-windows | disable | status ]

<table>
<thead>
<tr>
<th>port</th>
<th>serial port (eg, ttyS1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>local IP</td>
<td>local IP address</td>
</tr>
<tr>
<td>remote IP</td>
<td>remote IP address</td>
</tr>
<tr>
<td>-w</td>
<td>configure to run with a windows client</td>
</tr>
</tbody>
</table>
```

The `config-ppp` command is used to enable or disable the PPP server on Serial 1. When PPP is enabled, a PPP server listens for connections on the **Serial** 1 port.

The **dust** user can invoke the `config-ppp` command using `sudo`.

```
dust@manager$ sudo config-ppp enable
```
6.5 create-network-snapshot

This allows the user to create a snapshot of the current Manager and network state. The command is invoked as: `create-network-snapshot`

By default, this script creates a directory `/tmp/snapshot`. The snapshot is created as a tarball named `snapshot.tar.gz` in this directory. If there is an older `snapshot.tar.gz` file, it is moved to `snapshot.tar.gz.old`. 
### 6.6 dust-manager

The `dust-manager` script is the system-level init script for starting the Manager software. The `dust-manager` script can be invoked with `sudo` by the `dust` user. After performing its initialization, the `dust-manager` script calls the `mgrctl` script to perform the specified action.

```
dust@manager$ sudo /etc/init.d/dust-manager start
```

Standard actions:
- **start** – Start the manager software.
- **stop** – Stop the manager software.
- **restart** – Restart the manager software.

On the LTP5903CEN-WHR, the `dust-manager` script is responsible for setting several environment variables used by the Manager.

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Used by</th>
</tr>
</thead>
<tbody>
<tr>
<td>DUST_SERIAL_NUMBER</td>
<td>Used by the Manager software as the device id for licensing. Reported by the Manager API as part of the System element.</td>
</tr>
<tr>
<td>DUST_PRODUCT_NAME</td>
<td>Reported by the Manager API as part of the System element</td>
</tr>
<tr>
<td>DUST_HW_REVISION</td>
<td>Reported by the Manager API as part of the System element</td>
</tr>
<tr>
<td>DUST_DUSTMGR_REV</td>
<td>Reported by the Manager API as part of the System element</td>
</tr>
</tbody>
</table>
6.7 mgrctl

Startup and shutdown script for the Manager processes.

The `mgrctl` script is not the system-level init script provided on the device. The init script, `/etc/init.d/dust-manager`, performs additional operations such as setting required environment variables before calling the `mgrctl` script. It may require root privileges to run.

Standard actions:

- **start** – Start the manager software.
- **stop** – Stop the manager software.
- **restart** – Restart the manager software.
- **monitor** – Report whether the manager software is running.
- **promote** – Promote a slave to master.

The `mgrctl` script is also used for starting the Manager under a redundancy manager, such as Linux-HA (though no HA system is provided in this release). The script conforms to the Linux-HA OCF spec required by resource agents.
6.8 Network Configuration

A couple of scripts are provided for changing the system's network configuration. These scripts can be run under `sudo` by the `dust` user. usage: ifswitch-to-dhcp

Switch the network configuration to use DHCP to obtain IP address and DNS configuration.

```
dust@manager$ sudo ifswitch-to-dhcp
```

usage: ifswitch-to-static <ip address> [gateway address] [netmask] [dns server]

Switch the network configuration to use a static IP address. If the additional parameters are not provided, the script assumes default values based on a /24 subnet.

```
dust@manager$ sudo ifswitch-to-static 192.168.1.100
```
6.9 restore-dcc-conf

6.9.1 restore-dcc-conf

The `restore-dcc-conf` command stops the Manager and removes any persistent configuration, all ACL entries and log files, restoring the default shipping configuration.

dust@manager$ sudo restore-dcc-conf
Removing SMM log files...
Removing mote information...
Removing manager configuration...
Removing join configuration...
Removing user configuration...
Restoring /opt/dust-manager/conf/config/system.ini...
Restoring /opt/dust-manager/conf/settings/dcc.conf...
Restoring /opt/dust-manager/conf/settings/serial.conf...
Restoring /opt/dust-manager/conf/settings/xmlrpc.conf...
Restoring /opt/dust-manager/conf/settings/watchdog.conf...
Restoring /opt/dust-manager/conf/config/watchdog-client.conf...
6.10 restore-factory-conf

The `restore-factory-conf` command removes any persistent configuration of the device and restores the default shipping configuration, including user passwords for the Linux and Admin Toolset logins. The `restore-factory-conf` command also calls `restore-dcc-conf` to restore Manager configuration.

```
dust@manager$ sudo restore-factory-conf
```

Optionally, the `restore-factory-conf` command can be used to restore just the user passwords for both the Linux and Admin Toolset logins by passing the `--users` argument to the command. This command does not affect any other device configuration, nor the Manager configuration.

```
dust@manager$ sudo restore-factory-conf --users
Restoring default passwords
Restoring /etc/passwd...
Restoring /etc/shadow...
Restoring /opt/dust-manager/web/.htpasswd...
```

See Restoring Manager Factory Default Settings.
6.11 set-conf-param

6.11.1 set-conf-param

**usage:** set-conf-param <file> <parameter> [value]

Get or set a configuration parameter in a Manager ini file. This script only examines the existing configuration file. It does not know about default values compiled into the software. Therefore, if a file does not exist or a variable is not present in the configuration file, the script will not return a value.

For example:

```bash
dust@manager$ set-conf-param $SMARTMESH_HOME/conf/config/system.ini RDNCY_STANDALONE_MODE false
dust@manager$ set-conf-param $SMARTMESH_HOME/conf/config/system.ini RDNCY_STANDALONE_MODE true
```
6.12 set-time

usage: /usr/bin/set-time [-z <timeZone>] [-t <time>] [-s <server1> <server2> <server3>] [-q]

Set the system clock time. If the time is set manually, NTP synchronization will be disabled. If NTP servers are specified, NTP will be used to update the system time.

- **timeZone**: name of timezone to use
- **time**: current time in the format MMDDhhmm[[CC][YY]][.ss]
- **server**: hostname or IP of ntp timeservers to use
- **-q**: instead of prompting interactively for unspecified values, use existing configuration
Trademarks

Eterna, Mote-on-Chip, and SmartMesh IP, are trademarks of Dust Networks, Inc. The Dust Networks logo, Dust, Dust Networks, and SmartMesh are registered trademarks of Dust Networks, Inc. LT, LTC, LTM and are registered trademarks of Linear Technology Corp. All third-party brand and product names are the trademarks of their respective owners and are used solely for informational purposes.

Copyright

This documentation is protected by United States and international copyright and other intellectual and industrial property laws. It is solely owned by Linear Technology and its licensors and is distributed under a restrictive license. This product, or any portion thereof, may not be used, copied, modified, reverse assembled, reverse compiled, reverse engineered, distributed, or redistributed in any form by any means without the prior written authorization of Linear Technology.

RESTRICTED RIGHTS: Use, duplication, or disclosure by the U.S. Government is subject to restrictions of FAR 52.227-14(g) (2)(6/87) and FAR 52.227-19(6/87), or DFAR 252.227-7015 (b)(6/95) and DFAR 227.7202-3(a), and any and all similar and successor legislation and regulation.

Disclaimer

This documentation is provided “as is” without warranty of any kind, either expressed or implied, including but not limited to, the implied warranties of merchantability or fitness for a particular purpose.

This documentation might include technical inaccuracies or other errors. Corrections and improvements might be incorporated in new versions of the documentation.

Linear Technology does not assume any liability arising out of the application or use of any products or services and specifically disclaims any and all liability, including without limitation consequential or incidental damages.

Linear Technology products are not designed for use in life support appliances, devices, or other systems where malfunction can reasonably be expected to result in significant personal injury to the user, or as a critical component in any life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness. Linear Technology customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify and hold Linear Technology and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Linear Technology was negligent regarding the design or manufacture of its products.

Linear Technology reserves the right to make corrections, modifications, enhancements, improvements, and other changes to its products or services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to Dust Network’s terms and conditions of sale supplied at the time of order acknowledgment or sale.
Linear Technology does not warrant or represent that any license, either express or implied, is granted under any Linear Technology patent right, copyright, mask work right, or other Linear Technology intellectual property right relating to any combination, machine, or process in which Linear Technology products or services are used. Information published by Linear Technology regarding third-party products or services does not constitute a license from Linear Technology to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from Linear Technology under the patents or other intellectual property of Linear Technology.

Dust Networks, Inc is a wholly owned subsidiary of Linear Technology Corporation.

© Linear Technology Corp. 2012-2015 All Rights Reserved.