Connection manager provides the user interface and the supported commands are described in the subsequent sections.

Connection Manager supports the following features:

1. Wi-Fi Scan
2. Wi-Fi Association/Authentication
3. Wi-Fi Disassociation
4. Wi-Fi Disconnect Management.
5. Static/Dynamic IP set/get
6. Wi-Fi Connection status
7. Keep alive connection configuration
8. Keep alive start/stop
9. Keep alive connection TLS support
10. Host wakeup configuration (wake word) and clear
11. Auto scan enable/disable/configuration status
12. Shutdown command
13. Wi-Fi Power management set/get
14. FOTA configuration/start/stop
15. WLAN power management configuration
16. TOS parameter for network packets
17. Incoming/outgoing packet information display
18. Kill tunadapter
19. Wi-Fi provisioning over BLE

Connection Manager commands are in the following format:

|  |
| --- |
| conmgr <operation> <argument-1>….<argument-n> |

**Note**: *Arguments in [ ] are optional and those in <> are mandatory.*

## scan - Performs Wi-Fi Scan

|  |  |
| --- | --- |
| **Command** | conmgr scan |
| **Parameters** | NA |
| **Response** | Scan results as listed:   * BSSID * Channel * RSSI * Security * SSID |

Table 8: Scan - command, parameters and responses

Example: ./conmgr scan.

Text

Description automatically generated

Figure 5: ./conmgr scan – Wi-Fi scan

## connect – Connects to Wi-Fi Network

Enterprise security type: Personal

|  |  |
| --- | --- |
| **Command** | conmgr connect <ssid> <passphrase>[PMK\_PATH] |
| **Parameters** | *ssid*: Name of the Access Point  *passphrase*: Password (optional for open security)  *pmk\_path*: Path of the data file containing PMK cache,(optional, PMK cache works only on WPA2.) |
| **Response** | “Connected” to the mentioned AP. |

Table 9: Connect (personal) - command, parameters and responses

Example 1: Configure to connect to an Access Point of SSID innotest and passphrase 123456789:

|  |
| --- |
| ./conmgr connect innotest 123456789 |

A computer screen with white text

Description automatically generated

Figure 6: ./conmgr connect (Personal) – Connect to Wi-Fi network

Enterprise security type: EAP-PSK

|  |  |
| --- | --- |
| **Command** | conmgr connect <ssid> <passphrase><2> <identity> |
| **Parameters** | *ssid*: Name of the Access Point  *passphrase*: Password (optional for open security)  *2:* Network security type  *identity:* EAP username identity string |
| **Response** | “Connected” to the mentioned AP. |

Table 10: Connect (EAP-PSK) - command, parameters and responses

A screenshot of a computer

Description automatically generated

Figure 7: ./conmgr connect (EAP-PSK) – Connect to Wi-Fi network

Enterprise security type: EAP-TLS

|  |  |
| --- | --- |
| **Command** | conmgr connect <ssid> <3> <identity> <ca\_path> <cert\_path> <pkey\_path> <pkey\_pwd> |
| **Parameters** | *ssid*: Name of the Access Point  *3:* Network security type  *identity:* EAP username identity  *ca\_path:* CA certificate path in Talaria TWO filesystem  *cert\_path:* Client certificate path in Talaria TWO filesystem  *pkey\_path:* Path of private key file in Talaria TWO filesystem  *pkey\_pwd:* Private key password |
| **Response** | “Connected” to the mentioned AP. |

Table 11: Connect (EAP-TLS) – command, parameters and responses

A screenshot of a computer

Description automatically generated

Figure 8: ./conmgr connect (EAP-TLS) – Connect to Wi-Fi network

Enterprise security type: EAP-PEAP

|  |  |
| --- | --- |
| **Command** | conmgr connect <ssid> <4> <identity> <ca\_path> <identity2> <password><phase2auth> |
| **Parameters** | *ssid*: Name of the Access Point  *4:* Network security type  *identity:* EAP username identity  *ca\_path:* CA certificate path  *identity2:* EAP username identity for Phase 2  *password:* Private key password  *phase2auth:* Phase 2 authentication |
| **Response** | “Connected” to the mentioned AP. |

Table 12: Connect (EAP-PEAP) - command, parameters and responses

A screenshot of a computer

Description automatically generated

Figure 9: ./conmgr connect (EAP-PEAP) – Connect to Wi-Fi network

**Certificate Storing Method for Enterprise Security Certificates**

|  |
| --- |
| sudo openocd -s conf/ -f ftdi\_swd.cfg -f t2\_swd.cfg  sudo ./script/arden.py ./apps/gordon-jtag/bin/gordon-jtag.elf  sudo python3 ./script/gdbrun.py ./apps/gordon-jtag/bin/gordon-jtag.elf --noconsole –nowait  ./script/storage.py --device localhost:10000 write ~/Downloads/rootCACert.pem /data/ca.pem  ./script/storage.py --device localhost:10000 write ~/Downloads/client\_cert.pem /data/client.pem  ./script/storage.py --device localhost:10000 write ~/Downloads/client\_key.pem /data/client.key |

## disconnect – Disconnects from Wi-Fi Network

|  |  |
| --- | --- |
| **Command** | conmgr disconnect |
| **Parameters** | None |
| **Response** | “Disconnected” from the Wi-Fi network. |

Table 13: Disconnect - command, parameters and responses

Example: ./conmgr disconnect.

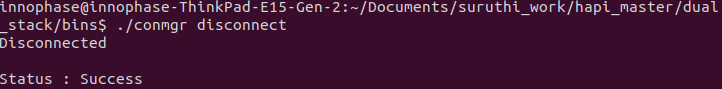


Figure 10: ./conmgr disconnect – Disconnect from Wi-Fi network

## IP – Get/Set ipv4 Address

|  |  |
| --- | --- |
| **Command** | conmgr ip <ipaddr4> <net mask> <gateway ip> <dns server ip> |
| **Parameters** | *ipaddr4*: IP address, as big-endian integer  *netmask*: Netmask, as big-endian integer  *gateway ip*: Default-route address, as big-endian integer.  *dns server ip*: DNS server address, as big-endian integer.  Execute the following to get current ipv4 address information:   |  | | --- | | conmgr ip | |
| **Response** | Gets the IP configuration status. |

Table 14: IP - command, parameters and responses

Example: Set static IP address 192.168.1.174, subnetmask 255.255.255.0, Gateway 192.168.1.1 and DNS serverip 192.168.1.1:

|  |
| --- |
| ./conmgr ip 192.168.1.174 255.255.255.0 192.168.1.1 192.168.1.1 |

Text

Description automatically generated

Figure 11: ./conmgr ip – Set/Get Static IP Address

## kaconfig – Configures keepalive/shadow Service

|  |  |
| --- | --- |
| **Command** | conmgr kaconfig <server ip > <server port> <interval> <heartbeat msg> <wakeup word> <timeout> |
| **Parameters** | *server ip*: Server IP address to start keep alive socket.  *server port*: Server port number.  *interval*: Interval to send heartbeat message.  *heartbeat msg*: Heartbeat message to be sent.  *wakeup word*: Wake word to receive from server. This wake word will be compared with data received from the server. If this matches, host wakeup will be triggered.  *timeout*: Timeout for heartbeat acknowledge message. |
| **Response** | Keepalive config: success message. |

Table 15: kaconfig - command, parameters and responses

Example: Send HeartBeatPacket message to server IP address 172.16.16.155 using port number 5000 every 3 seconds.

It times out if there has been no acknowledgment received from server after 10 seconds.

WakeUP is the string using which cloud can trigger Talaria TWO host wake up.

|  |
| --- |
| ./conmgr kaconfig 172.16.16.155 5000 3 HeartBeatPacket WakeUP 10 |

Text

Description automatically generated

Figure 12: kaconfig: Configure Keepalive Send

## kaconfigget – Gets keepalive/shadow Service Configuration

|  |  |
| --- | --- |
| **Command** | conmgr kaconfigget |
| **Parameters** | None |
| **Response** | This command provides the following information:   * association status: associated/not associated * keepalive configuration status: keepalive\_configured /keepalive not configured * keepalive status: Keepalive connected/ Keepalive Not connected” * keepalive interval: Interval in seconds to send keepalive message * keepalive port: Port number * keepalive IP: IP address * heartbeat msg: Keepalive message format * wakeup msg: Wakeup message format * heartbeat sent times: Number of times keepalive message was sent * wakeup recv times: Number of times wakeup message was received |

Table 16: kaconfigget - command, parameters and responses

Example: ./conmgr kaconfigget.

Text

Description automatically generated

Figure 13: kaconfigget - Read Keepalive Configuration

## kastart – Starts keepalive/shadow Service

|  |  |
| --- | --- |
| **Command** | conmgr kastart |
| **Parameters** | None |
| **Response** | Keepalive start: success/fail |

Table 17: kastart - command, parameters and responses

Example: ./conmgr kastart.

Text

Description automatically generated

Figure 14: kastart - Start Sending Keepalive to Server

## kastop – Stops keepalive/shadow Service

|  |  |
| --- | --- |
| **Command** | conmgr kastop |
| **Parameters** | None |
| **Response** | Keepalive stop: success/fail |

Table 18: kastop - command, parameters and responses

Example: ./conmgr kastop.

Text

Description automatically generated

Figure 15: kastop - Stop Sending Keepalive to Server

## kadel – Deletes keepalive/shadow Service Configuration

|  |  |
| --- | --- |
| **Command** | conmgr kadel |
| **Parameters** | None |
| **Response** | Keepalive conf delete: “success/failed” |

Table 19: kadel - command, parameters and responses

Example: ./conmgr kadel.

Text

Description automatically generated

Figure 16: kadel - Delete Keepalive Configuration

## status – Gets Dual-Stack Status

|  |  |
| --- | --- |
| **Command** | conmgr status |
| **Parameters** | None |
| **Response** | The status command respond contains following information:   * mode: STA. 0=STA * status: 1=associated/0=not associated * ssid: SSID of the connected network * bssid: BSSID of the connected network * sta-mac – station mac address * channel: Wi-Fi channel * rssi: Estimated RSSI for the station * IPv4 address: IP address of the connected network * subnet mask: Netmask, as big-endian integer * default gateway: default route, as big-endian integer * dns: DNS server, as big-endian integer * Security: WPA/WPA2 personal security * heap remaining: Heap memory remaining * Talaria TWO powersave: Power save status |

Table 20: status - command, parameters and responses

Example: ./conmgr status.

Text

Description automatically generated

Figure 17: status: Get Status Information

## wakeupconfig – Configures Host Wakeup PIN

|  |  |
| --- | --- |
| **Command** | conmgr wakeupconfig <wakeup pin> <wakeup level> <pin direction> |
| **Parameters** | *wakeup pin*: Wake up pin number  *wakeup level*: Host wakeup level (0/1)  *pin direction*: 0 – output (default) / 1 – input  **Note**: If no value is provided to *pin direction*, 0 is taken as default. |
| **Response** | Wakeup configuration: success/fail |

Table 21: wakeupconfig - command, parameters and responses

**Note**: To use wakeupconfig on a Host platform with MCU, GPIO14 and GPIO21 should be configured as described below:

1. *./conmgr wakeupconfig 14 1 1*

where,

* 1. wakeup pin=14
  2. wakeup level=1
  3. pin direction=1

1. *./conmgr wakeupconfig 21 0*

where,

* 1. wakeup pin=21
  2. wakeup level=0

Example: Configure gpio20 on Talaria TWO’s end as wakeup with high state.

|  |
| --- |
| ./conmgr wakeupconfig 20 1 |

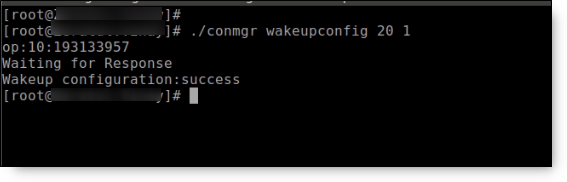


Figure 18: wakeupconfig - Configure Host Wakeup using GPIO

## getwakeupreason – Gets Host Wakeup Reason

|  |  |
| --- | --- |
| **Command** | conmgr getwakeupreason |
| **Parameters** | None |
| **Response** | Valid wakeup reasons are as follows:   * Network wakeup * TCP timeout * TCP fin * AP Disconnect * PIR * Button * Autoscan * DHCP renewal * Connected to network * Wakeup gpio1 * Wakeup gpio2 |

Table 22: getwakeupreason - command, parameters and responses

Example: ./conmgr getwakeupreason

Text

Description automatically generated

Figure 19: getwakeupreason - Get Wakeup Reason

## wakeupreasoncls – Clears Host Wakeup Reason

|  |  |
| --- | --- |
| **Command** | conmgr wakeupreasoncls |
| **Parameters** | None |
| **Response** | Wakeup reason clear: success |

Table 23: wakeupreasoncls - command, parameters and responses

Example: ./conmgr wakeupreasoncls

Text

Description automatically generated

Figure 20: wakeupreasoncls - Clear Wakeup Reason

## autoscanenable – Enables Autoscan Operation

|  |  |
| --- | --- |
| **Command** | conmgr autoscanenable<interval> <ssid> |
| **Parameters** | *interval*: Scan interval in seconds.  *ssid*: SSID to scan for. |
| **Response** | Autoscan Enabled/Autoscan Enable: failed |

Table 24: autoscanenable - command, parameters and responses

Example: Set the scan interval to 1 second for ssid ct\_asus.

|  |
| --- |
| ./conmgr autoscanenable 1 ct\_asus |

Text

Description automatically generated

Figure 21: autoscanenable - Enable Autoscan

## autoscandisable – Disables Autoscan Operation

|  |  |
| --- | --- |
| **Command** | conmgr autoscandisable |
| **Parameters** | None |
| **Response** | Autoscan Disabled/Autoscan Disable: failed |

Table 25: autoscandisable - command, parameters and responses

Example: ./conmgr autoscandisable.

Text

Description automatically generated

Figure 22: autoscandisable - Disable Autoscan

## shutdown – Triggers Host Shutdown

|  |  |
| --- | --- |
| **Command** | conmgr shutdown |
| **Parameters** | None |
| **Response** | Shutdown: success/failed |

Table 26: shutdown - command, parameters and responses

Example: ./conmgr shutdown.

Text

Description automatically generated

Figure 23: shutdown - Host shutdown

## kasecconfig – Configures Keepalive Security Parameters

|  |  |
| --- | --- |
| **Command** | conmgr kasecconfig <config> <certificate name> |
| **Parameters** | *config*: Configuration options:  1-> enable  0-> disable  *certificate name*: TLS certificate name |
| **Response** | Keep-alive sec config: success/failed |

Table 27: kasecconfig - command, parameters and responses

Example: Enable keepalive secure configuration with the certificate name cert.pem.

|  |
| --- |
| ./conmgr kasecconfig 1 cert.pem |

Text

Description automatically generated

Figure 24: kasecconfig - Enable Keepalive TLS

**Certificate storing method for kasecconfig:**

|  |
| --- |
| sudo openocd -s conf/ -f ftdi\_swd.cfg -f t2\_swd.cfg  sudo ./script/arden.py ./apps/gordon-jtag/bin/gordon-jtag.elf  sudo python3 ./script/gdbrun.py ./apps/gordon-jtag/bin/gordon-jtag.elf --noconsole –nowait  ./script/storage.py --device localhost:10000 write ~/Downloads/www-google-com.pem /data/cert/www-google-com.pem |

## wpmconfig – Configures WLAN Power Management

|  |  |
| --- | --- |
| **Command** | conmgr wpmconfig <listen\_interval> <traffic\_timeout> <pm\_options> |
| **Parameters** | *listen\_interval*: Listen to each listen\_interval beacon.  *traffic\_tmo*: Traffic timeout in milliseconds. The Wi-Fi interface will go to Wi-Fi power save if no traffic has occurred for this time.  *pm\_options* : Power save options:   1. ps-poll(1): Send ps poll if a beacon was missed. 2. Dynamic\_listen\_interval(2): Listen to each beacon if there has been traffic recently. 3. sta\_rx\_nap(4): Turn off receiver for uninteresting frames/station. 4. sta\_bcast\_only(8): Do not receive multicast frames that are not broadcasted. 5. tx\_ps(16): Send outgoing frames without leaving Wi-Fi power save 6. mcast\_ignore(32): Ignore the multicast flag in beacons. 7. dtim\_only(64): Listen to only DTIM beacons |
| **Response** | Talaria TWO wcm pm config: success/failed |

Table 28: wpmconfig - command, parameters and responses

Example: Configure the listen interval to 3, traffic timeout to 12ms and enable ps-poll.

|  |
| --- |
| ./conmgr wpmconfig 3 12 1 |

Text

Description automatically generated

Figure 25: wpmconfig – Configure WLAN power management

## powersave – Suspend Enable/Disable on Talaria TWO

|  |  |
| --- | --- |
| **Command** | conmgr powersave <Enable/Disable> |
| **Parameters** | Enable: 1  Disable:0 |
| **Response** | powersave: “success/failed” |

Table 29: powersave - command, parameters and responses

Example: Disable and enable suspend mode on Talaria TWO.

|  |
| --- |
| ./conmgr powersave 0  ./conmgr powersave 1 |

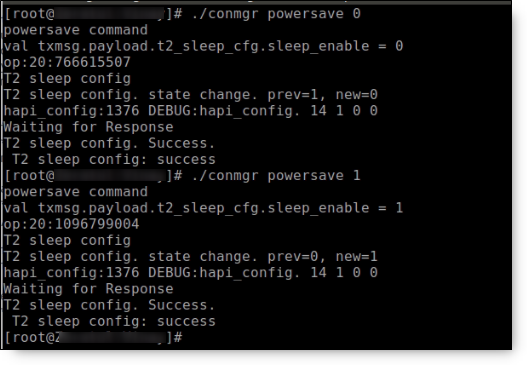


Figure 26: powersave – Suspend disable/enable on Talaria TWO

## version – Gets Version Information

|  |  |
| --- | --- |
| **Command** | conmgr version |
| **Parameters** | None |
| **Response** | version get: success/failed  Displays version information such as version of tunadapter, embedded app, conmgr , dual\_stack |

Table 30: version - command, parameters and responses

Example: ./conmgr version.

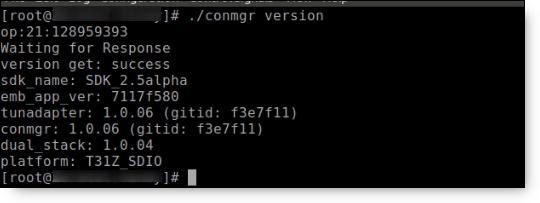


Figure 27: version - Get Version Information

## cbenable – Enable Connect/Disconnect Callback

|  |  |
| --- | --- |
| **Command** | conmgr cbenable |
| **Parameters** | None |
| **Response** | cbenable: “success/failed” |

Table 31: cbenable - command, parameters and responses

Example: ./conmgr cbenable.

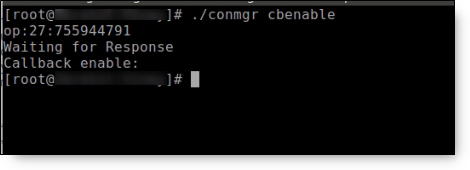


Figure 28: cbenable – Enable connect/disconnect callback

## fos – Upgrade the WLAN firmware from Host

|  |  |
| --- | --- |
| **Command** | conmgr <fos> <elf\_path> <hash> <auto\_reset> |
| **Parameters** | *elf\_path* (must): Path of the Dual-Stack ELF  *hash* (optional): sha256 hash of the elf provided through elf\_path. Use --no\_hash to skip this argument  *auto\_reset* (optional): 1 – Talaria TWO is reset automatically after Successful Upgrade  0 - Host needs to reset Talaria TWO after Successful Upgrade |
| **Response** | Firmware upgrade: “success/failed” |

Table 32: fos - command, parameters and responses

Example: Upgrading WLAN firmware from Host

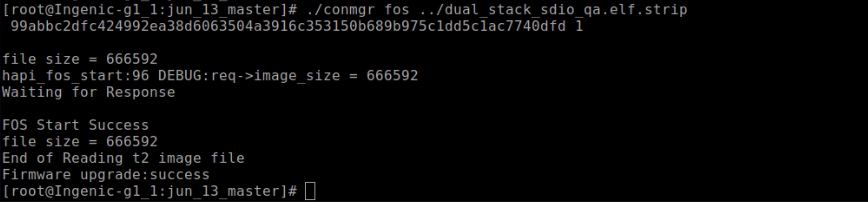


Figure 29: fos – Upgrade the WLAN firmware from Host

**Note**: Once the firmware upgrade is successfully completed, reset the host to enable the tunadapter to re-establish communication with Talaria TWO.

## hcidevname – Device Name

|  |  |
| --- | --- |
| **Command** | conmgr hcidevname |
| **Parameters** | None |
| **Response** | HCI device name is displayed  **Note**: Supported only on host platforms with SPI interface |

Table 33: hcidevname - command, parameters and responses

Example: ./conmgr hcidevname.

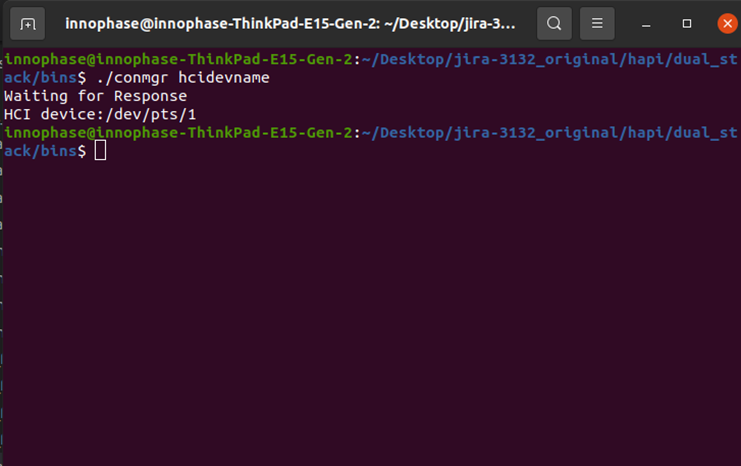


Figure 30: hcidevname – Device name

## kasecconfigget – Get Keepalive Security Parameters

Get secure configuration, the information of certificate name and enable security status will be resulted. If security is disabled (sec\_enable=0) then the certificate name will be NULL.

|  |  |
| --- | --- |
| **Command** | conmgr kasecconfigget |
| **Parameters** | None |
| **Response** | Keepalive security parameters are displayed followed by a success message.  **Note**: If security is disabled (sec\_enable=0), the certificate name will be NULL. |

Table 34: kasecconfigget - command, parameters and responses

Example: ./conmgr kaseccconfigget.

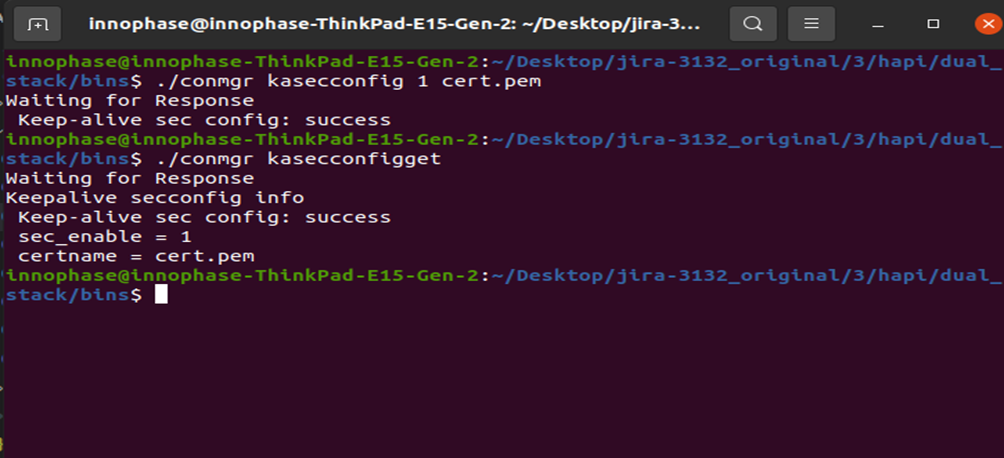


Figure 31: kasecconfigget - Get keepalive security parameters

## wakeupcfgget – Get Wakeup Configuration

|  |  |
| --- | --- |
| **Command** | conmgr wakeupcfgget |
| **Parameters** | None |
| **Response** | Wakeup configuration information followed by success message. |

Table 35: wakeupcfgget - command, parameters and responses

Example: ./ conmgr wakeupcfgget.

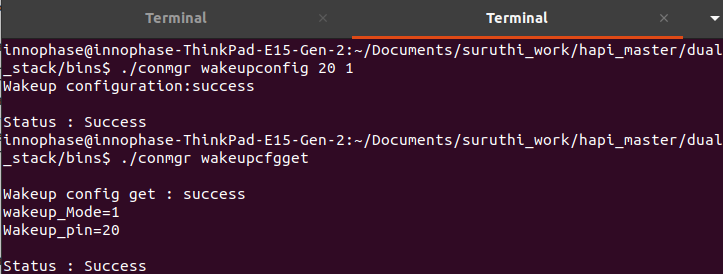


Figure 32: wakeupcfgget - Get wakeup configuration

## autoscancfgget – Get autoscan Configuration Status

|  |  |
| --- | --- |
| **Command** | conmgr autoscancfgget |
| **Parameters** | None |
| **Response** | Get autoscan status: success/fail message followed by the following parameters:   * interval: Scan interval in seconds * ssid: SSID to scan |

Table 36: autoscancfgget - command, parameters and responses

Example: ./conmgr autoscancfgget.

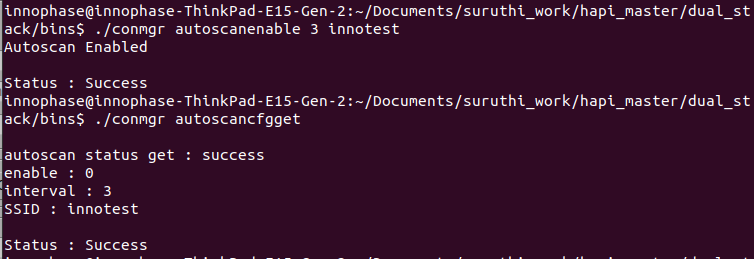


Figure 33: autoscancfgget - Get autoscan configuration status

## wpmconfigget – Get WLAN Power Management Configuration

|  |  |
| --- | --- |
| **Command** | conmgr wpmconfigget |
| **Parameters** | None |
| **Response** | Provides WLAN power management configuration such as:   * listen\_interval: Listen interval in units of beacon intervals. * traffic\_tmo: Traffic timeout in milliseconds. The Wi-Fi interface will go to Wi-Fi power save in case of no traffic for this duration. * pm\_options : Power save options:   + ps-poll(1): Send ps poll if a beacon was missed   + dynamic\_listen\_interval(2): Listen to each beacon if there has been traffic recently   + sta\_rx\_nap(4): Turn off receiver for uninteresting frames/station   + sta\_bcast\_only(8): Do not receive multicast frames that are not broadcasted   + tx\_ps(16): Send outgoing frames without leaving Wi-Fi power save   + mcast\_ignore(32): Ignore the multicast flag in beacons. |

Table 37: wpmconfigget- command, parameters and response

Example : ./ conmgr wpmconfigget.

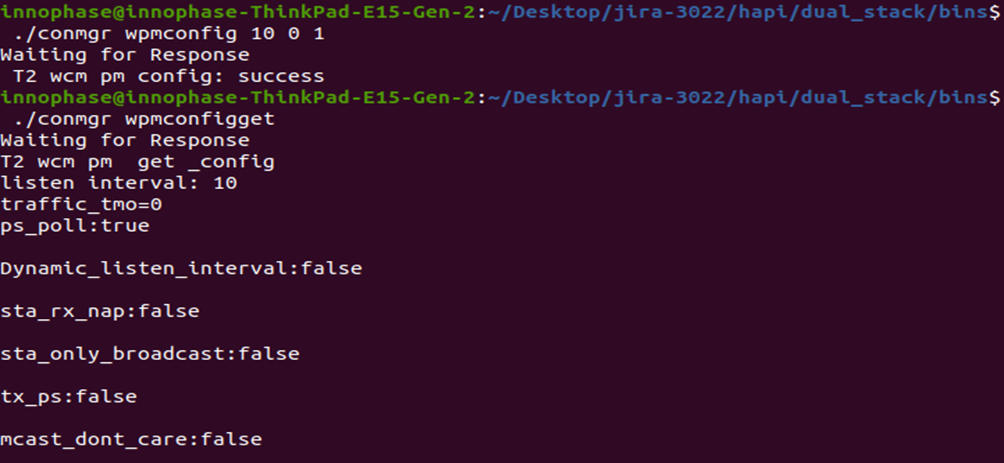


Figure 34: wpmconfigget - Get WLAN power management configuration

## tos – Set TOS Parameter for Network Packets

|  |  |
| --- | --- |
| **Command** | conmgr tos <tos\_value> |
| **Parameters** | tos\_value: Following are the values which can be used for setting tos:  tos=160 → Video  tos=192 → Voice  tos=0 → Best Effort  tos=32 → BK (Background) |
| **Response** | TOS set: success |

Table 38: tos - command, parameters and response

Example : ./ conmgr tos 160.



Figure 35: tos - Set tos parameter for network packets

## disp\_pkt\_info – Display Incoming and Outgoing Packet Information

|  |  |
| --- | --- |
| **Command** | conmgr disp\_pkt\_info<disp\_option> |
| **Parameters** | disp\_option: Following are the values for different display options:  0=Do not display  1=Display at Host only  2=Display at Talaria TWO only  3=Display at Host and Talaria TWO |
| **Response** | disp\_pkt\_info: success |

Table 39: disp\_pkt\_info - command, parameters and response

Example : ./ conmgr disp\_pkt\_info 0

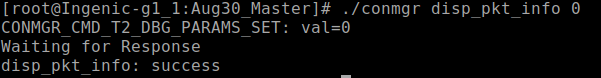


Figure 36: disp\_pkt\_info: Display incoming and outgoing packet information

## kill-tun – Gracefully Kills the Tunadapter

|  |  |
| --- | --- |
| **Command** | conmgr kill-tun |
| **Parameters** | None |
| **Response** | tun-kill: success |

Table 40: disp\_pkt\_info - command, parameters and response

Example : ./ conmgr kill-tun

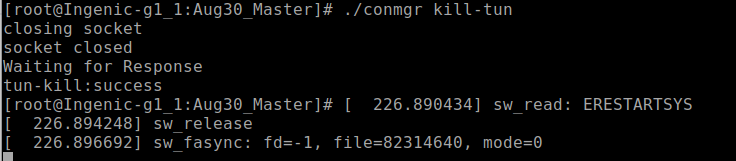


Figure 37: kill-tun: Gracefully kills the tunadapter

## provstart – Wi-Fi Provisioning over BLE

|  |  |
| --- | --- |
| **Command** | ./conmgr provstart <device\_name> <manufacturer\_name> |
| **Parameters** | *device\_name*: Name of the device for provisioning  *manufacturer\_name*: Device manufacturer name  **Note**:   * device\_name and manufacturer\_name are mandatory arguments, without providing which the BLE advertisement or scanning feature will not work. * If the device is connected to another Wi-Fi network previously, ensure it is disconnected from the network before running provstart. |
| **Response** | Success: Prov\_success  Failure: Prov\_fail |

Table 41: provstart - command, parameters and response

Once the provstart command is passed, the specified device begins advertising which can be observed using the BLE provisioning mobile application.

For more details on using the BLE provisioning mobile application, refer section: *Running the Application using Android or iOS app* from *Example\_using\_Provisioning.pdf* (*sdk\_x.y\examples\prov\doc*).

**Note**: x and y in sdk\_x.y refer to the SDK package release version.

Once provisioning is a success, the selected SSID and passphrase is displayed on the console. Pass the [./conmgr connect](#_connect_–_Connects) command with the same SSID and passphrase. Connection status can be determined by passing the [./conmgr status](#_status_–_Gets) command.

Example : ./ conmgr provstart Inno\_provisioning Innophase

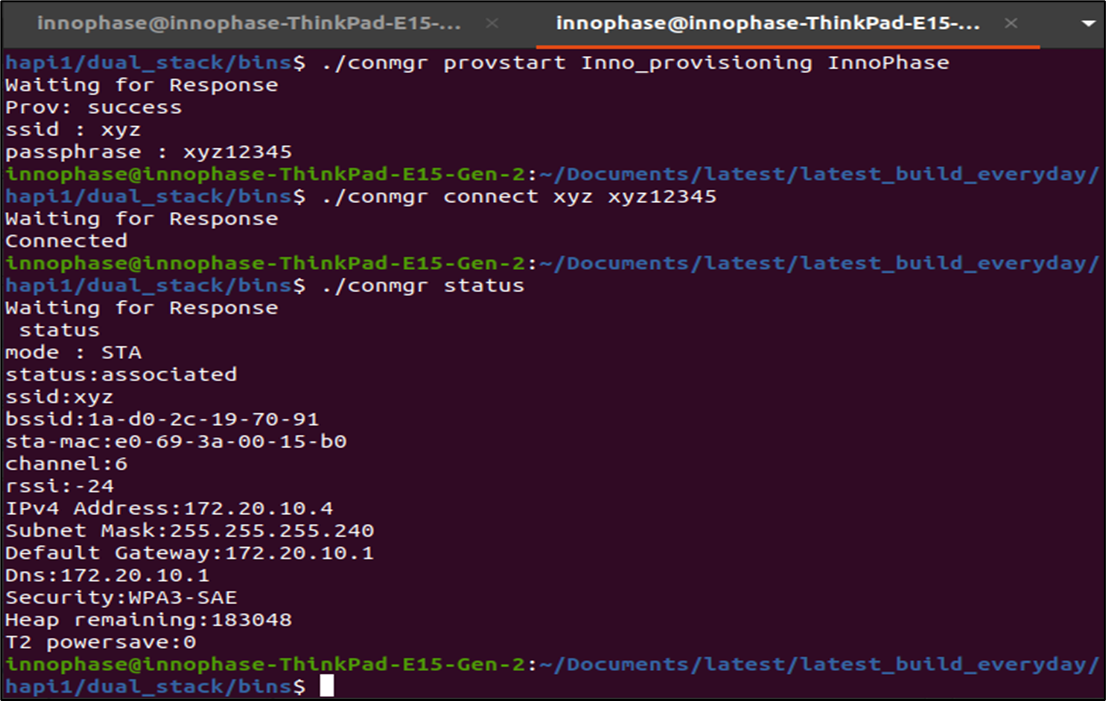


Figure 38: provstart – Wi-Fi provisioning over BLE

## add - Adds packet forward configuration and prints rule ID

|  |  |
| --- | --- |
| **Command** | conmgr pfcadd <ip\_type> <remote\_ip> <remote\_port> <remote\_port\_start> <remote\_port\_end> <local\_port><local\_port\_start> <local\_port\_end> <protocol> <direction> |
| **Parameters** | * ip type : IP type (0-ipv4, 1-ipv6) * remote\_ip : Remote IP address * remote\_port : Remote port address * remote\_port\_start : Remote port start address. Used to specify the port range * remote\_port\_end : Remote port end address. Used to specify the port range * local\_port : Local (source) port address * local\_port\_start : Local port start address. Used to specify the port range * local\_port\_end : Local port end address. Used to specify the port range * protocol : Protocol to apply the rule. Standard protocol values are used   (0-ignore protocol filter, 1- ICMP, 2- IGMP, 6-TCP, 17-UDP)   * direction : The direction to forward the packet.   Whether to Talaria TWO (0) or Host (1) |
| **Response** | rule-id: 6  Success/Failure |

Table 42: pfcadd - command, parameters and responses

Example: ./conmgr pfcadd 0 0 0 0 0 0 0 0 6 1

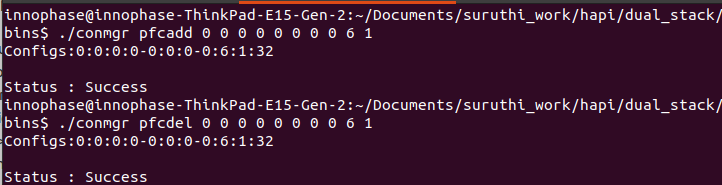


Figure 39: add - Adds packet forward configuration and prints rule ID

## del - Deletes packet forward configuration

|  |  |
| --- | --- |
| **Command** | conmgr pfcdel <ip\_type> <remote\_ip> <remote\_port> <remote\_port\_start> <remote\_port\_end> <local\_port> <local\_port\_start> <local\_port\_end> <protocol> <direction> |
| **Parameters** | * ip type : IP type (0-ipv4, 1-ipv6) * remote\_ip : Remote IP address * remote\_port : Remote port address * remote\_port\_start : Remote port start address. Used to specify the port range * remote\_port\_end : Remote port end address. Used to specify the port range * local\_port : Local (source) port address * local\_port\_start : Local port start address. Used to specify the port range * local\_port\_end : Local port end address. Used to specify the port range * protocol : Protocol to apply the rule. Standard protocol values are used   (0-ignore protocol filter, 1- ICMP, 2- IGMP, 6-TCP, 17-UDP)   * direction : The direction to forward the packet.   Whether to Talaria TWO(0) or host(1) |
| **Response** | Success/Failure |

Table 43: pfcdel - command, parameters and responses

Example: ./conmgr pfcdel 0 0 0 0 0 0 0 0 6 1

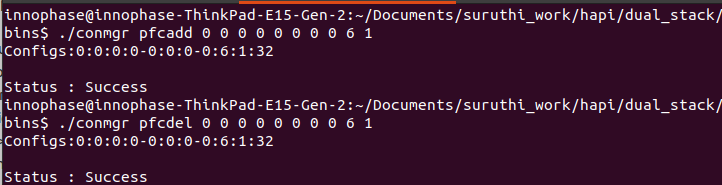


Figure 40: del - Deletes packet forward configuration

## del-id - Deletes packet forward configuration by providing rule ID

|  |  |
| --- | --- |
| **Command** | conmgr pfc-del-id <ip\_type> <rule\_id> |
| **Parameters** | * ip type : IP type (0-ipv4, 1-ipv6) * rule\_id : Rule ID to delete. This is provided by configuration **add** operation. |
| **Response** | Success/Failure |

Table 44: pfcdel-id - command, parameters and responses

Example: ./conmgr pfc-del-id 0 0

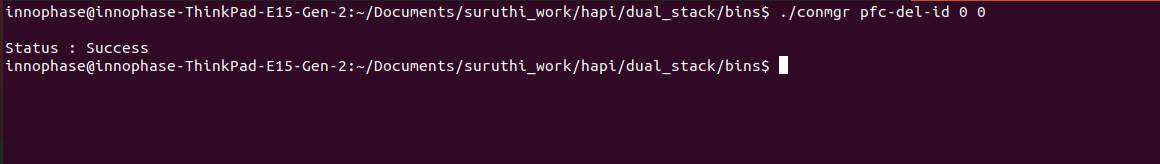


Figure 41: del-id - Deletes packet forward configuration by providing rule ID

## query - Prints forward configurations

|  |  |
| --- | --- |
| **Command** | conmgr pfcquery <ip\_type> |
| **Parameters** | 1. ip type : IP type (0-ipv4, 1-ipv6) |
| **Response** | Packet forward configurations  JSON format |

Table 45: pfcquery - command, parameters and responses

Example: ./conmgr pfcquery 0

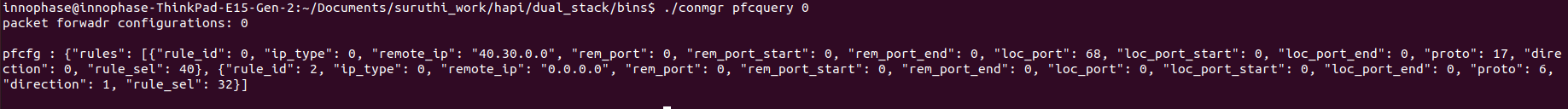


Figure 42: query - Prints forward configurations

## dirset/dirget - Sets/Gets packet forward direction

|  |  |
| --- | --- |
| **Command** | conmgr pfcdirset [direction]  conmgr pfcdirget |
| **Parameters** | * direction: The direction to forward the packet. Whether to Talaria TWO (0) or Host (1). This parameter is valid for GET operation. |
| **Response** | pfcdirset: Success/Failure  pfcdirget: Packet forwarding direction (Talaria TWO or Host) |

Table 46: pfcdir - command, parameters and responses

Example:

1. ./conmgr pfcdirset 0 ; ./conmgr pfcdirget
2. ./conmgr pfcdirset 1 ; ./conmgr pfcdirget

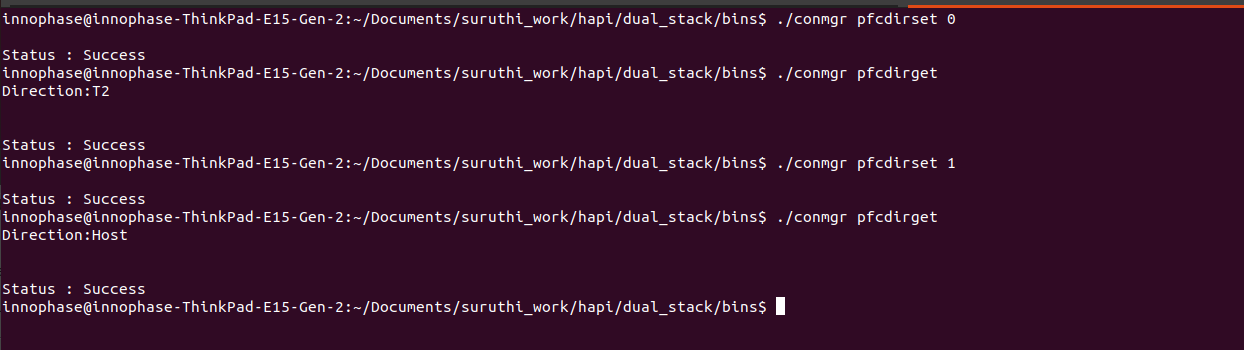


Figure 43: dirset/dirget - Sets/Gets packet forward direction