snickerdoodle

WIRELESS CONNECTION GUIDE

JANUARY 5, 2016

How to Read this Document

This document makes extensive use of links, references and notices in the page margins to detail additional information that can be useful while following the guide.

WARNING A warning notice indicates a potential hazard. If care is not taken to adhere to the safety precautions, damage may be done to snickerdoodle.

Warnings and cautions will be clearly visible in either the body of the text or in the margin and must be paid close attention while following the guided steps.

CAUTION A caution indication denotes a process that requires special attention. If the caution is not exercised and the process not adhered to, failure may result and/or potential damage to snickerdoodle.

Warning, caution and informational notices, such as this one, may also be found in the margin.

Keywords

Keywords and important terms are shown in *italicized* type. Additional important information can be found in the margins of text with superscript notation[1].

Navigation of menus and directories are shown using ***bold italicized*** type. Any hierarchical navigation is shown using an arrow to denote a ***Parent*** → ***child*** relationship.

`Teletype` text is used to highlight inputs, variables and system files within the host environment.

[1] Margin notes, such as this one, reference the body content and highlight technical details or references for further information.

Introduction

This document details the process and methods used to connect snickerdoodle to a local wireless network to establish network and internet connectivity. The files and commands outlined in this document are found accessed directly from a booted snickerdoodle using a terminal interface established through connection with the snickerdoodle serial port located at J1 or J2.

The following information contains details on how to connect to a variety of network security modes using a combination of automated and manual processes.

> *Portions of this document have been adapted from the TI WL18xx documentation found at http://processors.wiki.ti.com/index.php/Connect_to_Secure_AP_using_WPA_Supplicant*

WPA Supplicant (`wpa_supplicant`)

The `wpa_supplicant` is a program that initializes and establishes network connectivity based on a set of configurations specified by a combination of a configuration file and (optionally) command line input.

`wpa_supplicant.conf`

Network configuration can be stored and additionally recalled automatically when booting snickerdoodle to connect to known and trusted networks. These configurations are typically stored in `/etc/wpa_supplicant.conf`[2]. Multiple network configurations can be stored in a single `wpa_supplicant.conf`.

[2] Additional information on the structure of `wpa_supplicant.conf` can be found at http://linux.die.net/man/5/wpa_supplicant.conf

```
root@snickerdoodle:~$ wpa_supplicant -d -Dnl80211 -c/etc/wpa_supplicant.conf -iwlan0 -B
```

Configuration Structure

An example network configuration for an open (unsecured) network can be found below. The first three lines of the file specify the configuration for the front-end (`wpa_cli`). Additional configuration examples for various network security types can be found at the end of this document.

```
ctrl_interface=/var/run/wpa_supplicant
ctrl_interface_group=0
update_config=1

network={
        auth_alg=OPEN
        key_mgmt=NONE
        ssid="my_Network"
        mode=0
}
```

wpa_supplicant is daemon and therefore only one instance of it may be run on a snickerdoodle, all other modifications and configuration of settings must be made with the frontend application, wpa_cli.

WPA Command Line Interface (wpa_cli)

wpa_cli is a text-based frontend program for interacting with wpa_supplicant. It is used to query current status, change configuration, trigger events, and request interactive user input.

```
wpa_cli -i interface [-hv] [command ...]
```

-h prints help information

-v verbose output

-i interface interface name, typically wlan0

In all commands used in the examples the first argument is the interface that is being configured, preceded by the -i flag (-i wlan0).

wpa_cli Commands

Setting the attributes of a network is done through a set of commands that are specified as arguments to wpa_cli. Many of these commands require additional arguments to specify the attributes that are being set.

disconnect - Puts the interface into a disconnected state and waits for a reassociate command before connecting.

```
wpa_cli -i wlan0 disconnect
```

add_network - Adds and assigns an index number to a new network for the specified interface. All necessary configuration of the newly added network is done with set_network command.

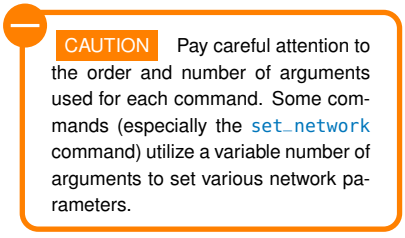
```
wpa_cli -i wlan0 add_network
```

select_network - Selects network with the specified succeeding index argument and disables others.

```
wpa_cli -i wlan0 select_network 0
```

enable_network - Enables network with specified succeeding index argument.

```
wpa_cli -i wlan0 enable_network 0
```



reassociate - Forces reassociation of the currently selected network.

```
wpa_cli -i wlan0 reassociate
```

status - Gets current WPA/EAPOL/EAP status of the currently selected network.

```
wpa_cli -i wlan0 status
```

list_networks - Lists configured networks for the specified interface.

```
wpa_cli -i wlan0 list_networks
```

remove_network - Removes network with the specified succeeding SSID argument from the list of configured networks.

```
wpa_cli -i wlan0 remove_network '"Bandipur"'
```

set_network Command

The following set of commands are used with the `set_network` command to specify network attributes. All of the following commands require the argument following "`set_network`" to be the network index, followed by any parameter commands and arguments.

auth_alg - Configures network authentication algorithm. Typically, the complimentary argument is `OPEN`.

```
wpa_cli -i wlan0 set_network 0 auth_alg OPEN
```

key_mgmt - Sets authenticated key management protocol for the network. Possible protocols are `NONE`, `WPA-EAP` or `WPA-PSK`.

```
wpa_cli -i wlan0 set_network 0 key_mgmt WPA-EAP
```

psk - Sets WPA passphrase or pre-shared key (PSK) to be used with the network.

```
wpa_cli -i wlan0 set_network 0 psk '"12345678"'
```

wep_key0 - Sets WEP key passphrase to be used by the network.

```
wpa_cli -i wlan0 set_network 0 wep_key0
    ABCdef1234567890abcDEF3333
```

pairwise - Sets CCMP algorithm as pairwise (unicast) cipher for WPA.

```
wpa_cli -i wlan0 set_network 0 pairwise CCMP
```

group - Sets CCMP algorithm as group (broadcast/multicast) cipher for WPA.

```
wpa_cli -i wlan0 set_network 0 group CCMP
```

If an error is made while setting a network parameter, simply re-input the corresponding `set_network` command with the correct arguments to override the parameter setting. To unset parameters (for different configuration types) the network must be removed from the list.

proto - Sets security protocol as WPA2.

```
wpa_cli -i wlan0 set_network 0 proto WPA2
```

eap - Sets PEAP as extensible authentication protocol's (EAP) method.

```
wpa_cli -i wlan0 set_network 0 eap PEAP
```

identity - Sets identity string for EAP.

```
wpa_cli -i wlan0 set_network 0 identity '"test"'
```

password - Sets password string for EAP.

```
wpa_cli -i wlan0 set_network 0 password '"test"'
```

phase1 - Sets outer authentication method version as PEAPv0.

```
wpa_cli -i wlan0 set_network 0 phase1 '"peapver=0"'
```

phase2 - Sets inner authentication method as EAP-MSCHAPv2.

```
wpa_cli -i wlan0 set_network 0 phase2 '"MSCHAPV2"'
```

mode - Sets IEEE 802.11 operation mode as infrastructure (Managed) mode, i.e., associate with an AP.

```
wpa_cli -i wlan0 set_network 0 mode 0
```

ssid - Sets network service set identifier (SSID) of the specified network index.

```
wpa_cli -i wlan0 set_network 0 ssid '"my_Network"'
```

`wpa_cli` Invocation

Configuring wireless networks using `wpa_cli` is an excellent way to test network connectivity and settings before applying those configurations to automatically establish connections. When using `wpa_cli`, it is important to invoke `wpa_supplicant` exactly ONCE before attempting to use any `wpa_cli` commands. `wpa_supplicant` should be used with the `-B` flag to daemonize the supplicant. An example call to `wpa_supplicant`:

```
wpa_supplicant -d -D nl80211 -c /etc/wpa_supplicant.conf -iwlan0 -B
```

This call uses the `nl80211` driver (specified by `-D`) and a configuration file specified by `-c`. The `-d` flag is used to output verbose debugging messages.

Removing Existing Networks

Removing all of the networks from the network list can be done using a combination of invocations of `wpa_cli`. This can be useful when wanting to work with only one network or testing network configurations manually. An example script/command which will remove all of the networks currently stored in the `wpa_cli` list is as follows:

```
for i in ‘wpa_cli -iwlan0 list_networks | grep ^[0-9] | cut -f1‘;
    do wpa_cli -iwlan0 remove_network $i; done
```

After removing all networks, a single network can be added by using the `add_network` command.

```
wpa_cli -i wlan0 add_network
```

Setting Network Parameters

After adding a network, it can be configured using a ’0’ index for any `set_network` commands that follow. Configuring the network parameters is done through a series of `set_network` commands that specify the characteristics of the network based on it’s authentication/security type and the credentials specific to the network. Below is an example of an unsecured (open) network. The `set_network` commands for other network configuration types can be found at the end of this document

```
wpa_cli -i wlan0 set_network 0 auth_alg OPEN
wpa_cli -i wlan0 set_network 0 key_mgmt NONE
wpa_cli -i wlan0 set_network 0 mode 0
wpa_cli -i wlan0 set_network 0 ssid ’"my_Network"’
```

After configuring the network using a series of `set_network` commands, the network can be enabled using the following series of commands:

```
wpa_cli -i wlan0 select_network 0
wpa_cli -i wlan0 enable_network 0
wpa_cli -i wlan0 reassociate
```

To verify connection status with the WPA supplicant, the `status` command can be used as shown below:

```
root@snickerdoodle:~$ wpa_cli -i wlan0 status
bssid=ff:ee:dd:cc:bb:aa
ssid=my_Network
id=0
mode=station
pairwise_cipher=CCMP
group_cipher=CCMP
key_mgmt=WPA2-PSK
wpa_state=COMPLETED
p2p_device_address=12:34:56:78:90:ab
address=cd:ef:11:22:33:44
```

Additional information about the wireless connection can be viewed using the `iw` command.

```
root@snickerdoodle:~$ iw wlan0 link
Connected to ff:ee:dd:cc:bb:aa (on wlan0)
        SSID: my_Network
        freq: 2462
        RX: 216135 bytes (735 packets)
        TX: 1345 bytes (13 packets)
        signal: -39 dBm
        tx bitrate: 72.2 MBit/s MCS 7 short GI

        bss flags:      short-preamble short-slot-time
        dtim period:    1
        beacon int:     100
```

For networks where a DHCP server is already running and administering IP addresses to clients, the `dhclient` command can be used to obtain an IP address and finalize connection to the network. Using `ifconfig` will confirm the snickerdoodle IP address.

```
root@snickerdoodle:~$ dhclient wlan0
root@snickerdoodle:~$ ifconfig
wlan0     Link encap:Ethernet  HWaddr 12:34:56:78:90:ab
          inet addr:10.0.111.112  Bcast:10.0.111.255
              Mask:255.255.255.0
          inet6 addr: fe80::36b1:f7ff:fedf:6f93/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1130 errors:0 dropped:1 overruns:0 frame:0
          TX packets:15 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:305537 (305.5 KB)  TX bytes:2134 (2.1 KB)
```

wpa_supplicant.conf Network Configurations

This section contains a set of example network configurations for `wpa_supplicant.conf` with various security types. Common network security types have been chosen to be included here, however, they are not the only security schemes that are possible. The `wpa_supplicant` man page contains some additional examples and helpful resources for further reading.

WPA

```
network={
        auth_alg=OPEN
        key_mgmt=WPA-PSK
        psk="1234abcdWXYZ"
        ssid="MYNETWORK"
        mode=0
}
```

WPA2

```
network={
        auth_alg=OPEN
        key_mgmt=WPA-PSK
        psk '"1234abcdWXYX"
        ssid="my_Network"
        proto=RSN
        mode=0
}
```

WEP 40 Open

```
network={
        auth_alg=OPEN
        wep_key0=1234567890
        key_mgmt=NONE
        ssid="my_Network"
        mode=0
}
```

WEP 128 Open

```
network={
        auth_alg=OPEN
        wep_key0=1234abcdWXYZ5678efghSTUV90
        key_mgmt=NONE
        ssid="my_Network"
        mode=0
}
```

WPA PSK

```
network={
        auth_alg=OPEN
        key_mgmt=WPA-PSK
        psk="1234abcdWXYZ"
        pairwise=CCMP TKIP
        group=CCMP TKIP
        ssid="my_Network"
        mode=0
}
```

`wpa_cli` Network Configurations

This section contains sets of commands used to configure network settings using `wpa_cli`. Each of the commands are variants of the `set_network` command and should be preceded by commands required to remove or add networks to the network list and followed by commands used to check the status of and finalize the connection, as described in the previous sections.

WPA

```
wpa_cli -i wlan0 set_network 0 auth_alg OPEN
wpa_cli -i wlan0 set_network 0 key_mgmt WPA-PSK
wpa_cli -i wlan0 set_network 0 psk '"12345678"'
wpa_cli -i wlan0 set_network 0 mode 0
wpa_cli -i wlan0 set_network 0 ssid '"my_Network"'
```

WPA2

```
wpa_cli -i wlan0 set_network 0 auth_alg OPEN
wpa_cli -i wlan0 set_network 0 key_mgmt WPA-PSK
wpa_cli -i wlan0 set_network 0 psk '"12345678"'
wpa_cli -i wlan0 set_network 0 proto RSN
wpa_cli -i wlan0 set_network 0 mode 0
wpa_cli -i wlan0 set_network 0 ssid '"my_Network"'
```

WPA/WPA2 PSK

```
wpa_cli -i wlan0 set_network 0 auth_alg OPEN
wpa_cli -i wlan0 set_network 0 key_mgmt WPA-PSK
wpa_cli -i wlan0 set_network 0 psk '"12345678"'
wpa_cli -i wlan0 set_network 0 pairwise CCMP TKIP
wpa_cli -i wlan0 set_network 0 group CCMP TKIP
wpa_cli -i wlan0 set_network 0 mode 0
wpa_cli -i wlan0 set_network 0 ssid '"my_Network"'
```

WEP 40 Open

```
wpa_cli -i wlan0 set_network 0 auth_alg OPEN
wpa_cli -i wlan0 set_network 0 wep_key0 1234567890
wpa_cli -i wlan0 set_network 0 key_mgmt NONE
wpa_cli -i wlan0 set_network 0 mode 0
wpa_cli -i wlan0 set_network 0 ssid '"my_Network"'
```

WEP 128 Open

```
wpa_cli -i wlan0 set_network 0 auth_alg OPEN
wpa_cli -i wlan0 set_network 0 wep_key0 1234abcdWXYZ5678efghSTUV90
wpa_cli -i wlan0 set_network 0 key_mgmt NONE
wpa_cli -i wlan0 set_network 0 mode 0
wpa_cli -i wlan0 set_network 0 ssid '"my_Network"'
```

WPA Enterprise (EAP TLS)

```
wpa_cli -i wlan0 set_network 0 auth_alg OPEN
wpa_cli -i wlan0 set_network 0 key_mgmt WPA-EAP
wpa_cli -i wlan0 set_network 0 pairwise TKIP
wpa_cli -i wlan0 set_network 0 group TKIP
wpa_cli -i wlan0 set_network 0 proto WPA
wpa_cli -i wlan0 set_network 0 eap TLS
wpa_cli -i wlan0 set_network 0 identity '"test"'
wpa_cli -i wlan0 set_network 0 client_cert '"/etc/certs/cert.pem"'
wpa_cli -i wlan0 set_network 0 private_key '"/etc/certs/key.pem"'
wpa_cli -i wlan0 set_network 0 private_key_passwd '"test"'
wpa_cli -i wlan0 set_network 0 mode 0
wpa_cli -i wlan0 set_network 0 ssid '"my_Network"'
```

WPA Enterprise (EAP PEAP0)

```
wpa_cli -i wlan0 set_network 0 auth_alg OPEN
wpa_cli -i wlan0 set_network 0 key_mgmt WPA-EAP
wpa_cli -i wlan0 set_network 0 pairwise TKIP
wpa_cli -i wlan0 set_network 0 group TKIP
wpa_cli -i wlan0 set_network 0 proto WPA
wpa_cli -i wlan0 set_network 0 eap PEAP
wpa_cli -i wlan0 set_network 0 identity "test"
wpa_cli -i wlan0 set_network 0 password "test"
wpa_cli -i wlan0 set_network 0 phase1 "peapver=0"
wpa_cli -i wlan0 set_network 0 phase2 "MSCHAPV2"
wpa_cli -i wlan0 set_network 0 mode 0
wpa_cli -i wlan0 set_network 0 ssid '"my_Network"'
```

WPA2 Enterprise (EAP TLS)

```
wpa_cli -i wlan0 set_network 0 proactive_key_caching 1
wpa_cli -i wlan0 set_network 0 auth_alg OPEN
wpa_cli -i wlan0 set_network 0 key_mgmt WPA-EAP
wpa_cli -i wlan0 set_network 0 pairwise CCMP
wpa_cli -i wlan0 set_network 0 group CCMP
wpa_cli -i wlan0 set_network 0 proto WPA2
wpa_cli -i wlan0 set_network 0 eap TLS
wpa_cli -i wlan0 set_network 0 identity '"test"'
wpa_cli -i wlan0 set_network 0 client_cert '"/etc/certs/cert.pem"'
wpa_cli -i wlan0 set_network 0 private_key '"/etc/certs/key.pem"'
wpa_cli -i wlan0 set_network 0 private_key_passwd '"test"'
wpa_cli -i wlan0 set_network 0 mode 0
wpa_cli -i wlan0 set_network 0 ssid '"my_Network"'
```

WPA2 Enterprise (EAP PEAP0)

```
wpa_cli -i wlan0 set_network 0 auth_alg OPEN
wpa_cli -i wlan0 set_network 0 key_mgmt WPA-EAP
wpa_cli -i wlan0 set_network 0 pairwise CCMP
wpa_cli -i wlan0 set_network 0 group CCMP
wpa_cli -i wlan0 set_network 0 proto WPA2
wpa_cli -i wlan0 set_network 0 eap PEAP
wpa_cli -i wlan0 set_network 0 identity '"test"'
wpa_cli -i wlan0 set_network 0 password '"test"'
wpa_cli -i wlan0 set_network 0 phase1 '"peapver=0"'
wpa_cli -i wlan0 set_network 0 phase2 '"MSCHAPV2"'
wpa_cli -i wlan0 set_network 0 mode 0
wpa_cli -i wlan0 set_network 0 ssid '"my_Network"'
```

WPA/WPA2 Enterprise (EAP TLS)

```
wpa_cli -i wlan0 set_network 0 auth_alg OPEN
wpa_cli -i wlan0 set_network 0 key_mgmt WPA-EAP
wpa_cli -i wlan0 set_network 0 proto WPA2
wpa_cli -i wlan0 set_network 0 eap TLS
wpa_cli -i wlan0 set_network 0 identity '"test"'
wpa_cli -i wlan0 set_network 0 client_cert '"/etc/certs/cert.pem"'
wpa_cli -i wlan0 set_network 0 private_key '"/etc/certs/key.pem"'
wpa_cli -i wlan0 set_network 0 private_key_passwd '"test"'
wpa_cli -i wlan0 set_network 0 mode 0
wpa_cli -i wlan0 set_network 0 ssid '"my_Network"'
```