V-MAC Usage Guide

Mohammed Elbadry

1 Introduction

This document details the procedure to install *V-MAC* kernel module on linux and remove TCP/IP and 802.11 standard mac layer. The procedure requires specific kernel version and custom built loadable kernel modules to complete.

1.1 Requirements

- Raspberry Pi 2, Pi 3, Pi 3+, Pi 4, or Nvidia Jetson
- SoftMAC Wi-Fi dongle, we recommend AlfaAWUS036NHA (List of supported devices)
- Loadable custom kernel modules
- Linux 4.14.68-v7+, 4.14.50-v7+, 5.0, or 4.9.x

2 Installing raspbian (skip if different platform)

2.1 Pi 2,3

The system is developed on 4.14.68 but has been moderately tested on 4.14.50 to ease installation. To install raspbian 4.14.50, please refer to archive and download .zip (i.e. image) under folder "raspbian-2018-06-29". An easy installation tool can be found at the following link .

2.2 Pi 4

Please download and install the image from the following link

2.3 Changing Firmware

Please swap *htc_9271-1.4.0.fw* located in */lib/firmware/ath9k_htc* with one in package then reboot once before loading the kernel modules.
Note: This step needs to be done only one time.

3 Loading Kernel Module

Please ensure there is only one softmac dongle connected to the raspberry pi at the moment after booting the device. Please unzip the VMAC folder given. Then use the following command to adjust permissions:

```
chmod 775 install.sh
./install.sh
```

Please use command *dmesg* to ensure the module is installed by seeing log messages like the following: "VMAC: Device registered and ready to transmit".

4 API

V-MAC API consists of few function calls, one being callback function (reception function, please refer to 4.3). *vmac_register* registers the process with the kernel module, and *send_vmac* that allows transmitting data-centric frames through V-MAC.

4.1 *v-mac_register(void * ptr)*

Registers process with the kernel module and takes pointer of callback function for reception (ref 4.3).
Note: Upon this function call, vmac kernel module prints a message that can be accessed through *dmesg* indicating Process ID registration success. Please refer to function *recv_frame* to see parameters for reception function parameters.

4.2 *int send_vmac(struct vmac_frame, struct meta_data)*

vmac_frame : structure that contains payload and data name with their lengths, respectively.
meta_data : structure containing type of frame, frame rate, and sequence number.

4.3 *recv_frame(struct vmac_frame, struct meta_data)*

vmac_frame : structure that contains payload and data name with their lengths, respectively.
meta_data : structure containing type of frame, frame rate, and sequence number.

5 Sample Code

A sample is included in the package with file named *stress-test.c* to compile the file, please use the following command:

```
gcc stress-test.c csiphash.c vmac-usrsp.c -pthread -lm
```

Note: You must remove entry from hashmap after receiving packet fully through the API. Furthermore, please do ensure freeing all memory of received information (i.e. that includes interest name if one exists).

6 Frame Injection

Frame injection allows sending frame with specific sequence number, thus allows 100% reliability through network layer without overhead of transmitting the whole data packet (i.e. fine-grained robustness per-frame level).

7 Frame Rate Selection

There is a lookup table located in *vmac-usrsp.h*, please refer to the lookup table of all available rates to select from it whichever rate you would like to use. The available rates span from 1Mbps (BPSK) to 150Mbps (64-QAM, HT40 with SGI).