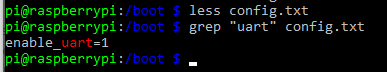
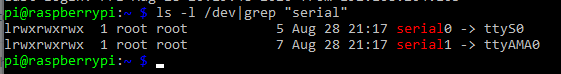
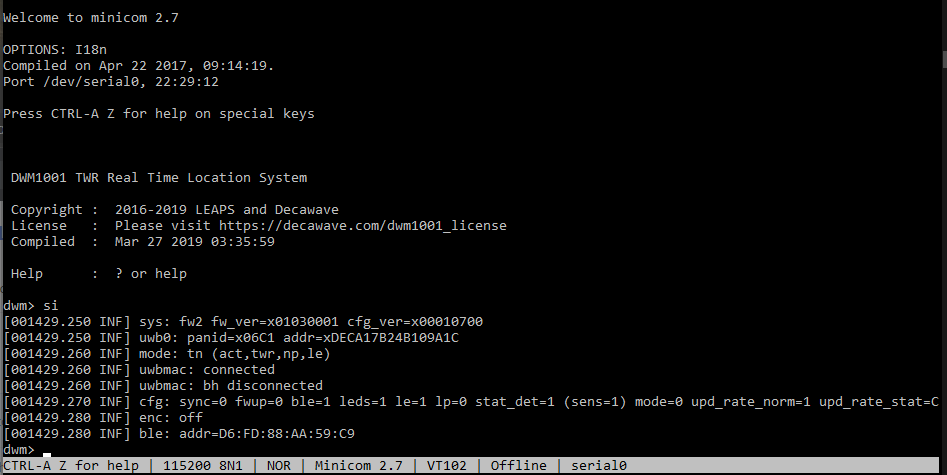
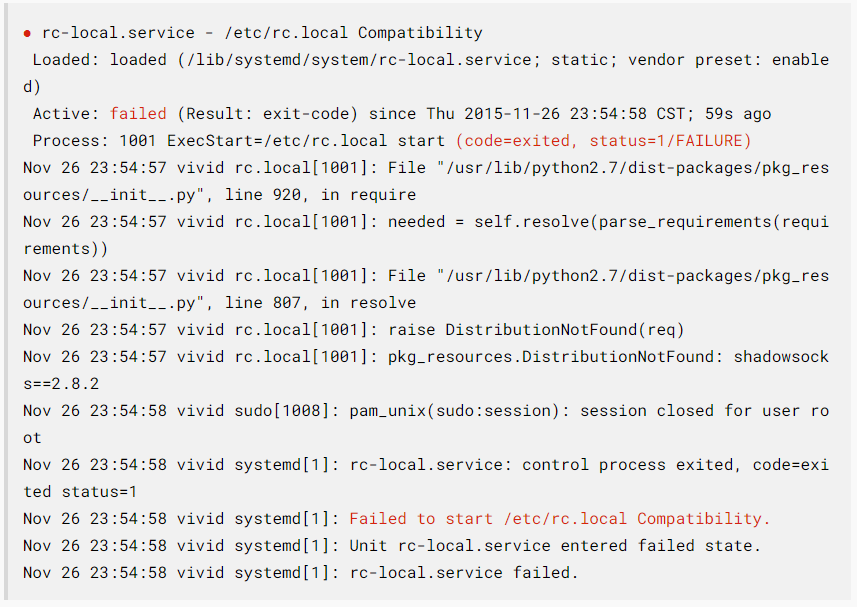
# **Instruction to convert Decawave 1000C into Tag with publishing capabilities**

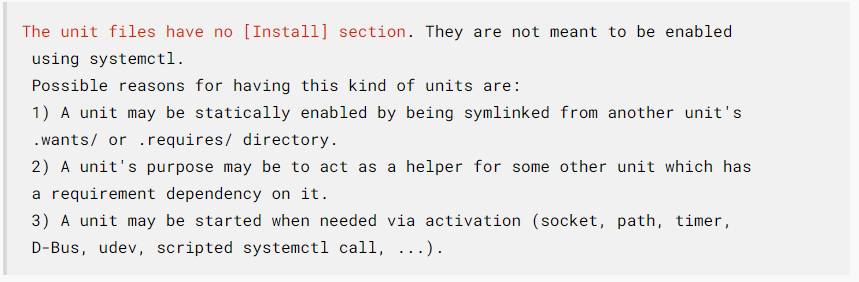
* Hardware requirement
  + DWM 1000C
  + Raspberry Pi 3 or above with Wi-Fi capabilities
  + Display, keyboard and mouse (for initial setup). You can also use ssh terminal if can make your Raspberry Pi to connect to Wi-Fi with boot up.
* Instructions
  + Make sure OS image on Raspberry Pi is 4.19.66-v7+. To check use
    - uname -a
  + Update and upgrade system’s package
    - sudo apt-get update
    - sudo apt-get dist-upgrade
  + Install below required package
    - Minicom - to get GPIO access
      * sudo apt-get install minicom
    - Mosquitto MQTT Broker - to publish data
      * sudo apt-get install mosquitto
    - Install git to fetch latest file
      * sudo apt install git-all
    - Check the python version. Python 3.5 and above is required.
      * python3 –version
    - Reboot the device either by UI or by below command
      * sudo reboot
  + Now setup is divided in three parts
    - Enabling UART for GPIO in Raspberry Pi.
    - Getting updated shell scripts, python scripts and config files from git [repository](https://github.com/hegxiten/UWB-MATLAB-Lab.git).
    - Modifying start up script for Raspberry pi to execute publishing python code on start up.
  + Setting up UART for GPIO
    - Enable UART GPIO
      * Modify config.txt at /boot/config.txt to add enable\_uart =1
        + sudo vi /boot/config.txt
        + add **enable\_uart=1**



* Check where serial ports are pointing
  + ls -l /dev|grep “serial”



* By default, Raspberry Pi uses the serial port for “console” login since we will we using GPIO pins to get data from Decawave chip using UART we will have to disable the console.
  + sudo systemctl stop serial-getty@ttyS0.service
  + sudo systemctl disable serial-getty@ttyS0.service
  + remove “**console=serial0,115200**” from cmdline.txt at /boot/cmdline.txt
    - sudo vi /boot/cmdline.txt and remove “**console=serial0,115200**”
  + verify that Raspberry Pi can fetch data from DWM1000
    - sudo minicom -b 115200 -D /dev/serial0
    - press enter twice once you get dwm prompt type si
  + Copy latest scripts and config files
    - Create a directory with name “git” inside /home/pi and clone master branch from repository inside git
      * cd ~/
      * sudo mkdir git
      * cd git
      * sudo git clone <repo name>
    - Copy tag\_pub\_service.sh, tag\_mqtt\_publisher.py from ~/git/UWB-MATLAB-Lab/UWB-Experiments-MATLAB/MQTT\_Communication\_Scripts to ~/ and make the .sh file executable.
      * mv tag\_mqtt\_publisher.py tag\_pub\_service.sh ~/
      * chmod +x tag\_pub\_service.sh
    - Similarly Copy rc.local from ~/git/UWB-MATLAB-Lab/UWB-Experiments-MATLAB/MQTT\_Communication\_Scripts to /etc/ and make it executable.
  + Modifying start up script for Raspberry pi to execute publishing python code on start up.
    - Latest rc.local file copied will have modification for start-up script.
    - Activate rc.local services to make tag work with reboot itself
      * Check status of rc-local if inactive enable the service
        + sudo systemctl status rc-local
      * Enable rc-local. If installation failed, then update rc-local .service.
        + sudo systemctl enable rc-local



* + - * + Update rc-local.service at /etc/systemd/system/ with below content

[Unit]

Description=/etc/rc.local Compatibility

ConditionPathExists=/etc/rc.local

[Service]

Type=forking

ExecStart=/etc/rc.local start

TimeoutSec=0

StandardOutput=tty

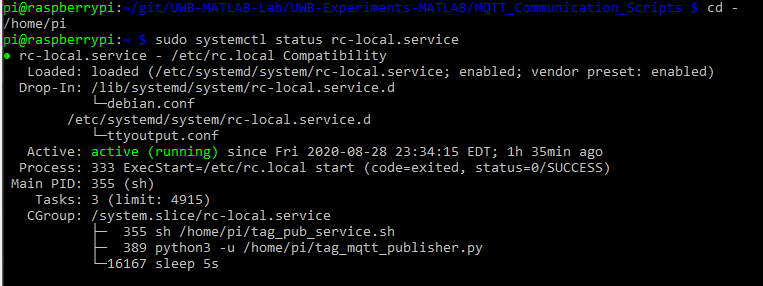
RemainAfterExit=yes

SysVStartPriority=99

[Install]

WantedBy=multi-user.target

* + - * Enable rc-local with same command as above. Start rc-local.services
        + sudo systemctl start rc-local.service
      * Reboot and check status again
        + reboot
        + sudo systemctl status rc-local.service



* Now note the WLAN IP address of Raspberry pi to connect to. Reboot the system and use IP address to connect to publisher in MQTT.fx and verify the process.

Data Collection Script (MATLAB) Runtime Setup:

The data collection script for the MQTT-based Wi-Fi backbone data link specifically requires the following tool(s) to execute:

1. Nmap

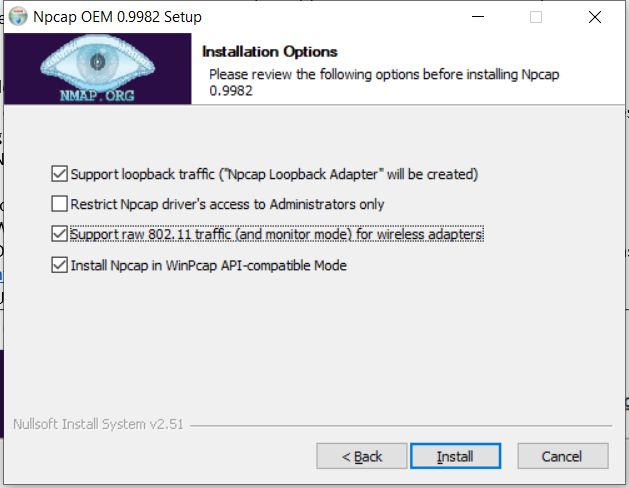
Special notice of installation:

1. Windows/PC platform:

Download the executable installation file of nmap from the official website (Self-installer)

<https://nmap.org/book/inst-windows.html#inst-win-exe>

**Uncheck the option “Restrict Npcap driver’s access to Administrator only”**



Restart MATLAB before after installation.

1. MAC OS platform:

Install through homebrew:

Execute terminal command:

“brew install nmap”

(if brew is not available, install brew by executing:

ruby -e "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)" < /dev/null 2> /dev/null

)

1. Linux platform – follow the official guide from nmap

This document will continuously be updated till whole process is automated and we no longer require this document.