

Ham Radio 'How-To' Guide

PAT Winlink – Raspberry Pi - Digirig

Mike Spohn

N1SPW

Revision: V.1.0

2025-08-09

www.n1spw.net

Copyright © 2025

"If it works out of the box – what fun is that?"

Table of Contents

A Note to New Hams and Non-Techies.....	4
A Note to Linux Gurus and Impatient Types.....	4
Introduction.....	5
'How-To' Goal.....	5
Requirements.....	5
Step-1: Hardware.....	6
Step-2: Software.....	8
Step-3: Configuration.....	16
Step-4: Radio/Comms.....	20
Step-5: Operations.....	27
Wrap-Up.....	28
Appendix I: Troubleshooting.....	29
Appendix II: Radio Settings.....	31
Appendix III: Installation Script.....	33
Appendix IV: Pat-Winlink Startup Script.....	34

Table of Figures

Figure 1: Digirig audio device ID.....	7
Figure 2: Digirig alternate device name.....	7
Figure 3: Digirig USB device assignment.....	8
Figure 4: Winlink Express.....	8
Figure 5: Winlink Express Setup.....	9
Figure 6: System patch.....	10
Figure 7: Install build-essential.....	10
Figure 8: Install cmake.....	10
Figure 9: Install git.....	11
Figure 10: Install libasound2-dev.....	11
Figure 11: Install libudev-dev.....	11
Figure 12: Install libavahi-client-dev.....	12
Figure 13: Install Midori.....	12
Figure 14: Install ax25-tools.....	12
Figure 15: git clone direwolf.....	13
Figure 16: cmake.....	13
Figure 17: make.....	14
Figure 18: make install.....	14
Figure 19: make install-conf.....	14
Figure 20: wget PAT-Winlink.....	15
Figure 21: Install PAT-Winlink package.....	15
Figure 22: arecord -l.....	16
Figure 23: ADEVICE setting.....	16
Figure 24: MYCALL setting.....	17
Figure 25: MODEM setting.....	17
Figure 26: List USB devices.....	17
Figure 27: PTT setting.....	18
Figure 28: PAT-Winlink configuration.....	18
Figure 29: AX.25 ports.....	19

Figure 30: wl2k port.....	19
Figure 31: Direwolf console.....	20
Figure 32: Alsa Mixer devices.....	21
Figure 33: Alsa Mixer sound settings.....	21
Figure 34: Kissatach.....	22
Figure 35: pat http.....	22
Figure 36: PAT-Winlink web application.....	22
Figure 37: Compose Email.....	23
Figure 38: Telnet connect.....	23
Figure 39: Telnet console log.....	24
Figure 40: All systems go.....	24
Figure 41: Compose Email.....	25
Figure 42: Email Outbox.....	25
Figure 43: Send Email.....	26
Figure 44: start_winlink.desktop.....	34

A Note to New Hams and Non-Techies

If you are a new Ham licensee – congratulations and welcome to the world’s greatest hobby. Those of you that already are Hams and do not have a lot of technical skills, the “How-To” series of documents was created just for you.

The technical side of Ham radio operations is very exciting. Pairing up modern computer technology with analog and digital Ham radio signals is an incredible achievement. It did not evolve, however, with the human in mind. In order to make a radio send digital data requires a number of individual components that must all be configured to work together. If one of these components is missing, or mis-configured, the system will simply not work.

This is what is so frustrating to new Hams and non-techies. The requirement to understand how each of the components work, and how to get them to work together is often a nightmare.

As you begin your journey to explore the technical side of computers and Ham radio, I urge you to be patient and not get frustrated. When you cannot get your system to work after hours of trying, you will probably think, “This is nuts!”. You are absolutely right. Remember, *modern tech was not designed with the human in mind*.

If you are willing to take the journey into the technical side of Ham radio, I promise you will be rewarded. You will accomplish things you never thought a Ham radio could do, and you will learn – a lot. As you progress, you will get even hungrier for more knowledge. And the cycle continues.

I have experienced the frustration. There were times I felt like throwing my radio out the window. Yet I stuck with it. I did not give up. I asked for help from the more experienced. I ‘squelched’ my frustration. Over time, I gained a level of expertise that surprised me.

I remember the hard times. I want to make things easier for you.

I strive to create documentation that is clear, concise, and walks through every step of implementing technology that is new to you. Each document has lots of screen-shots, diagrams, process flows, and checklists. They were written to make you successful.

A Note to Linux Gurus and Impatient Types

If you know your way around Linux or just want to get your Winlink station up and running fast, I wrote a bash script that installs all of the software and required dependencies. You can learn more about it in Appendix III.

There are also instructions on how to create a Desktop shortcut to start Pat-Winlink with two-clicks. That is described in Appendix IV.

NOTE: If you use the install script, be sure to add an entry to the `/etc/ax25/axports` file (See page 18).

Introduction

This document is a member of the **N1SPW** ‘How-To Series’ of documents, created to fill the gap between *YouTube* videos and real life. I applaud all of my Ham colleagues that have created hundreds of great videos, visualizing how to accomplish some cool Ham technology hack.

Videos are great, but in most cases, they do not provide enough detail to get a system up and running. What seems to work without a hitch in the video, often becomes a nightmare when an average Ham attempts to recreate what they saw the presenter do.

To ease the frustration of “*YouTube* implementation pain” experienced by so many of us, my contribution to the Ham community is some long overdue documentation.

‘How-To’ Goal

The goal of this ‘How-To’ is to document the steps needed to send Emails on FM radio channels using PAT-Winlink, a Raspberry Pi, and a Digirig. In short, I will show how to configure a Raspberry Pi to send Winlink Emails using a Digirig digital audio device and a Baofeng UV-5R H/T. I chose these components because I want to open the world of Winlink to you using the smallest budget possible.

A secondary goal is to push new Hams and non-techies outside their tech comfort zone to learn new things.

Requirements

To send Emails via PAT-Winlink, there are a lot of parts that have to work together to achieve success. I have broken these parts down into five categories:

1. Hardware
2. Software
3. Configuration
4. Radio/Comms
5. Operations

Hardware

The hardware you need includes the following:

- ✓ A Raspberry Pi (Pi 3, 4 or 5)
- ✓ Digirig Mobile
- ✓ Digirig cable for your specific radio
- ✓ USB-C cable that works with a Digirig

Software

The software you need includes the following:

- ✓ System software dependencies
- ✓ Pat-Winlink (Open-source Winlink client for Linux)
- ✓ Direwolf (Open-source software TNC)

Radio/Comms

For the radio will need the following:

- ✓ An FM radio transceiver (e.g. Baofeng UV-5R)

- ✓ A Winlink FM RMS (Gateway) within range
- ✓ Clean RF signal between your radio and the RMS

You must ensure all of the above play nice together if you want to send Emails over radio waves.

Step-1: Hardware

Raspberry Pi (Model 3, 4 or 5)

The first thing you need is a Raspberry Pi running a current version of Pi OS. For this “How-To”, I used a Pi model 3B V 1.2. If you are adventurous and want to try to do this with a Raspberry Pi Zero, it does work. I tried it. You have to run it “headless” (No GUI). In an upcoming “How-To” I will show how to configure a Pi Zero to send Winlink Emails.

There are plenty of documents and videos on the Internet to get you up and running with a Pi. I burned a 16GB micro SD card with the 32 bit [Pi OS image](#) with desktop dated March 13, 2025 (Kernel 6.12). I booted the Pi with the SD Card and performed the following steps once booted.

- Performed system initialization task as prompted.
- Ran the command from a terminal: `$ sudo raspi-config`
 - Under the System option I changed the hostname to “pat-winlink”.
 - Under the Interfaces option I enabled SSH and VNC
- Ran the below command from a terminal:
 - `$sudo apt update && sudo apt upgrade -y`
- Rebooted the Pi

Digirig

You can purchase a Digirig on [Amazon](#). I suggest you order one directly from the Digirig [web store](#); they are cheaper and *Digirig* does not have to pay Amazon's, fees so it keeps costs down. When you place your order, pay attention to the different CAT configurations. For most modern radios, you should select “Logic levels (default)”.

Digirig cable for your radio

You also need to buy or make a *Digirig* cable for your radio. You can find one for your radio at the [Digirig store](#). I use this black [Baofeng cable](#) for my UV-5R.

NOTE: Many of *Digirig's* cables work in multiple radios. For example, the cable kit for the Baofeng radios will work with my *Anytone AT-D878UV* radio. In fact, most radio's with the K1 connector (two prongs) will work with this cable kit. This is also known as a “Kenwood” cable.

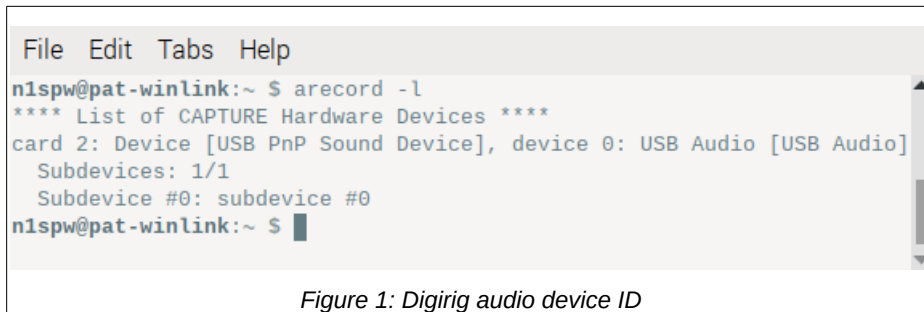
USB-C cable that works with a *Digirig*

It is *critically* important that you find a USB-C cable that works correctly with a Digirig. Not all USB-C cables are the same. You can buy excellent [USB cables](#) from Digirig that have chokes on each end. If you have troubles getting your Digirig to work – the first thing to suspect is your USB cable.

Configure the Digirig

There is not much to do to “configure” a Digirig. We need to find out two things: 1) What audio device ID is assigned to the Digirig, and 2) What USB port is assigned. Plug the USB cable into the Digirig, then plug the other end into your Pi.

To determine the audio device ID, run the below command from a terminal window on your Pi:
`$ arecord -l`

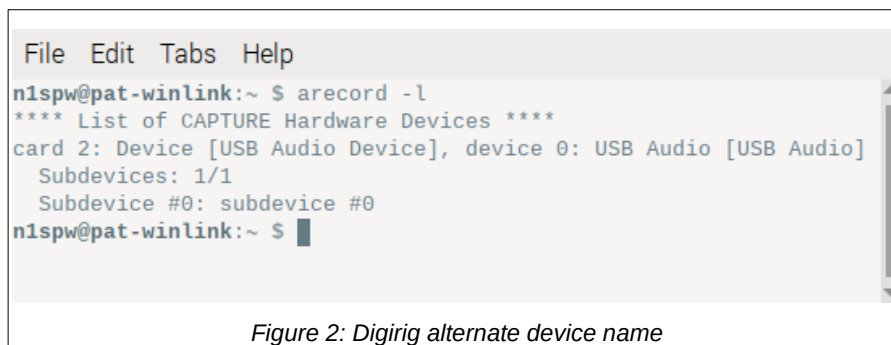


```
File Edit Tabs Help
n1spw@pat-winlink:~ $ arecord -l
**** List of CAPTURE Hardware Devices ****
card 2: Device [USB PnP Sound Device], device 0: USB Audio [USB Audio]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
n1spw@pat-winlink:~ $
```

Figure 1: Digirig audio device ID

You are specifically looking for an audio device named: “USB Audio Device”. In Figure-1 above, you can see my older Digirig ID is 2, 0 (card-2, device-0).

Make a note of this ID, you will need it later when you configure the software.



```
File Edit Tabs Help
n1spw@pat-winlink:~ $ arecord -l
**** List of CAPTURE Hardware Devices ****
card 0: Device [USB Audio Device], device 0: USB Audio [USB Audio]
  Subdevices: 1/1
  Subdevice #0: subdevice #0
n1spw@pat-winlink:~ $
```

Figure 2: Digirig alternate device name

Figure-2 above shows the device listing when I plug in a newer Digirig. Notice the device ID is the same, but the device name is different (USB Audio Device).

If you have more than one audio device in the list, you can unplug the Digirig and run the `$ arecord -l` command again. The device assigned to the Digirig will no longer be shown.


NOTE: You can always identify the Digirig by its unique name, “USB Audio Device”.

Next, we need to determine what USB device is assigned to the Digirig. To do so, enter the below command from a terminal window.

```
\ls -l /dev/ttyUSB*
```

NOTE: The `\` in front of the `ls` command removes the font background color from the listing, making it easier to see.

This command will list all USB devices connected to the system.



```
File Edit Tabs Help
n1spw@pat-winlink:~ $ \ls -l /dev/ttyUSB*
crw-rw---- 1 root dialout 188, 0 Jun 30 10:07 /dev/ttyUSB0
n1spw@pat-winlink:~ $
```

Figure 3: Digirig USB device assignment

The Digirig is the only USB device connected to my Pi. You can see it in Figure-3. It has been assigned *ttyUSB0*. If you see multiple USB devices in your list, you can determine which device is your Digirig by unplugging it from the Pi. The USB device assigned to it will disappear from the listing.

Make a note of this device ID, you will need it later when you configure the software.

Step-2: Software

Winlink Express

You need to create a Winlink account if you do not already have one. The easiest way to do this is to download and install [Winlink Express](#) on a Windows computer. Once installed, run Winlink Express.

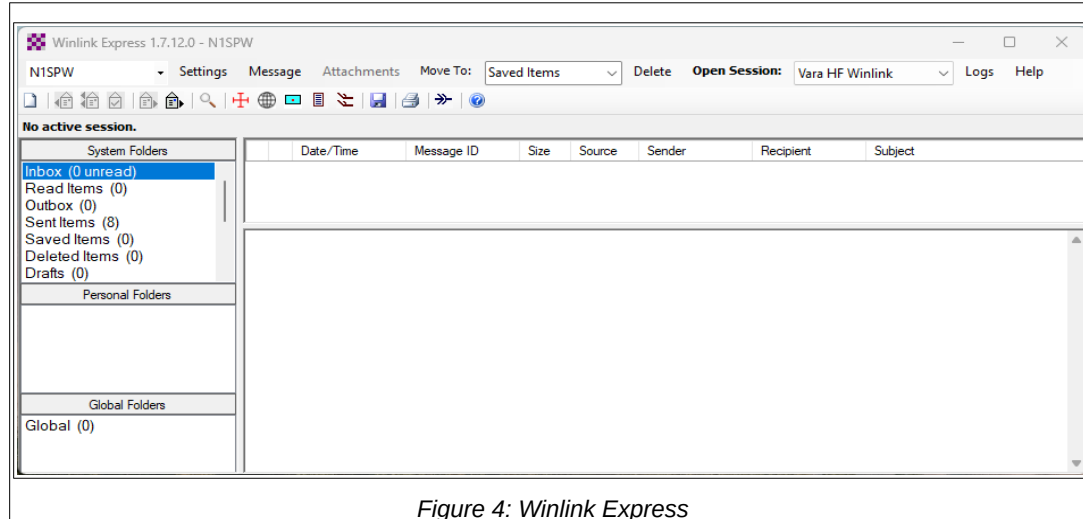


Figure 4: Winlink Express

Next, create a Winlink account. This can be done on the 'Settings' page. Click on the 'Settings' menu, and select 'Winlink Express Setup' (Figure 5).

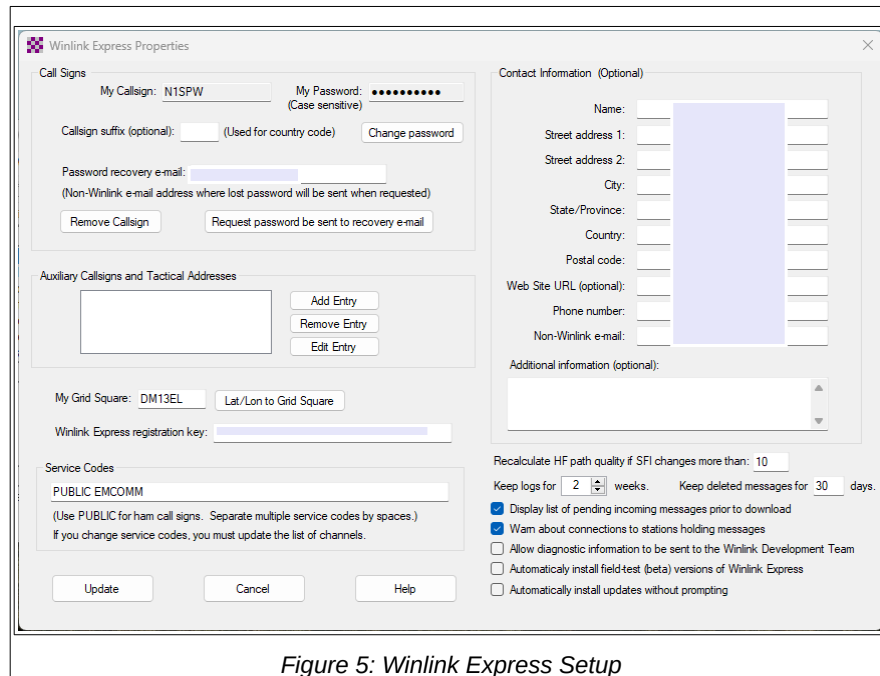


Figure 5: Winlink Express Setup

Fill in the required information:

- Call sign
- Password
- Password recovery Email address
- Grid square
- Service code(s)
- Contact information

Click 'Update'. Your account will be created once verification of your Call sign is completed. Your Winlink Email address is your <call sign>@winlink.org. Winlink will send you an Email with your Winlink password. I strongly suggest you change it to something only you know.

I also suggest you read the Winlink documentation to get familiar with the program. At a minimum, you should open a 'Telnet Winlink' session and practice sending Emails to/from your Winlink account and your personal Email account via the Internet.

NOTE: The rest of the required software will be installed on your Pi. For those of you that do not want to do this manually, I created a shell script that will install all the software. This script is included in the zip file that contains this document. More details can be found in Appendix III.

Required dependencies

The two main software components we need are Direwolf and Pat-Winlink. Direwolf is a software TNC (Terminal node controller) that manages the transfer of AX.25 signals to/from your radio. Pat-Winlink is browser-based software that performs all of the Winlink Email functions.

Prior to installing Direwolf and PAT-Winlink, we need to install some software they depend upon.

Boot up your Pi and open a terminal window.

Run the below commands:

```
$ sudo apt update && sudo apt upgrade -y
```

```
File Edit Tabs Help
n1spw@pat-winlink:~ $ sudo apt update && sudo apt upgrade -y
Hit:1 http://raspbian.raspberrypi.com/raspbian bookworm InRelease
Hit:2 http://archive.raspberrypi.com/debian bookworm InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
All packages are up to date.
W: http://raspbian.raspberrypi.com/raspbian/dists/bookworm/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Calculating upgrade... Done
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
n1spw@pat-winlink:~ $
```

Figure 6: System patch

```
$ sudo apt install build-essential -y
```

```
File Edit Tabs Help
n1spw@pat-winlink:~ $ sudo apt install build-essential
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  build-essential
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 7,704 B of archives.
After this operation, 20.5 kB of additional disk space will be used.
Get:1 http://mirror.fcix.net/raspbian/raspbian bookworm/main armhf build-essential armhf 12.9 [7,704 B]
Fetched 7,704 B in 1s (8,108 B/s)
Selecting previously unselected package build-essential.
(Reading database ... 142390 files and directories currently installed.)
Preparing to unpack .../build-essential_12.9_armhf.deb ...
Unpacking build-essential (12.9) ...
Setting up build-essential (12.9) ...
n1spw@pat-winlink:~ $
```

Figure 7: Install build-essential

```
$ sudo apt install cmake -y
```

```
File Edit Tabs Help
n1spw@pat-winlink:~ $ sudo apt install cmake
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  cmake-doc cmake-format elpa-cmake-mode
The following NEW packages will be installed:
  cmake
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 3,859 kB of archives.
After this operation, 23.3 MB of additional disk space will be used.
Get:1 http://mirror.fcix.net/raspbian/raspbian bookworm/main armhf cmake armhf 3.25.1-1 [3,859 kB]
Fetched 3,859 kB in 2s (2,164 kB/s)
Selecting previously unselected package cmake.
(Reading database ... 142386 files and directories currently installed.)
Preparing to unpack .../cmake_3.25.1-1_armhf.deb ...
Unpacking cmake (3.25.1-1) ...
Setting up cmake (3.25.1-1) ...
Processing triggers for man-db (2.11.2-2) ...
n1spw@pat-winlink:~ $
```

Figure 8: Install cmake

\$ sudo apt install git -y

```
File Edit Tabs Help
n1spw@pat-winlink:~ $ sudo apt install git
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  git-daemon-run | git-daemon-sysvinit git-doc git-email git-gui gitk gitweb git-cvs git-mediawiki
  git-svn
The following NEW packages will be installed:
  git
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 6,158 kB of archives.
After this operation, 50.2 MB of additional disk space will be used.
Get:1 http://raspbian.raspberrypi.com/raspbian bookworm/main armhf git armhf 1:2.39.5-0+deb12u2 [6,158
kB]
Fetched 6,158 kB in 2s (2,732 kB/s)
Selecting previously unselected package git.
(Reading database ... 141503 files and directories currently installed.)
Preparing to unpack .../git_1%3a2.39.5-0+deb12u2_armhf.deb ...
Unpacking git (1:2.39.5-0+deb12u2) ...
Setting up git (1:2.39.5-0+deb12u2) ...
n1spw@pat-winlink:~ $
```

Figure 9: Install git

\$ sudo apt install libasound2-dev -y

```
File Edit Tabs Help
n1spw@pat-winlink:~ $ sudo apt install libasound2-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  libasound2-doc
The following NEW packages will be installed:
  libasound2-dev
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 110 kB of archives.
After this operation, 676 kB of additional disk space will be used.
Get:1 http://archive.raspberrypi.com/debian bookworm/main armhf libasound2-dev armhf 1.2.8-1+rpt1 [110
kB]
Fetched 110 kB in 1s (98.6 kB/s)
Selecting previously unselected package libasound2-dev:armhf.
(Reading database ... 142319 files and directories currently installed.)
Preparing to unpack .../libasound2-dev_1.2.8-1+rpt1_armhf.deb ...
Unpacking libasound2-dev:armhf (1.2.8-1+rpt1) ...
Setting up libasound2-dev:armhf (1.2.8-1+rpt1) ...
n1spw@pat-winlink:~ $
```

Figure 10: Install libasound2-dev

\$ sudo apt install libudev-dev -y

```
File Edit Tabs Help
n1spw@pat-winlink:~ $ sudo apt install libudev-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  libudev-dev
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 53.0 kB of archives.
After this operation, 145 kB of additional disk space will be used.
Get:1 http://raspbian.raspberrypi.com/raspbian bookworm/main armhf libudev-dev armhf 252.36-1~deb12u1+
rpi1 [53.0 kB]
Fetched 53.0 kB in 1s (68.5 kB/s)
Selecting previously unselected package libudev-dev:armhf.
(Reading database ... 142323 files and directories currently installed.)
Preparing to unpack .../libudev-dev_252.36-1~deb12u1+rpi1_armhf.deb ...
Unpacking libudev-dev:armhf (252.36-1~deb12u1+rpi1) ...
Setting up libudev-dev:armhf (252.36-1~deb12u1+rpi1) ...
Processing triggers for man-db (2.11.2-2) ...
n1spw@pat-winlink:~ $
```

Figure 11: Install libudev-dev

\$ sudo apt install libavahi-client-dev -y

```
File Edit Tabs Help
n1spw@pat-winlink:~ $ sudo apt install libavahi-client-dev
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  libavahi-client-dev
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 49.2 kB of archives.
After this operation, 180 kB of additional disk space will be used.
Get:1 http://raspbian.raspberrypi.com/raspbian bookworm/main armhf libavahi-client-dev armhf 0.8-10+deb12u1 [49.2 kB]
Fetched 49.2 kB in 1s (66.3 kB/s)
Selecting previously unselected package libavahi-client-dev:armhf.
(Reading database ... 142384 files and directories currently installed.)
Preparing to unpack .../libavahi-client-dev_0.8-10+deb12u1_armhf.deb ...
Unpacking libavahi-client-dev:armhf (0.8-10+deb12u1) ...
Setting up libavahi-client-dev:armhf (0.8-10+deb12u1) ...
n1spw@pat-winlink:~ $
```

Figure 12: Install libavahi-client-dev

\$ sudo apt sudo apt install midori -y

```
File Edit Tabs Help
n1spw@pat-winlink:~ $ sudo apt install midori -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following NEW packages will be installed:
  midori
0 upgraded, 1 newly installed, 0 to remove and 0 not upgraded.
Need to get 705 kB of archives.
After this operation, 4,053 kB of additional disk space will be used.
Get:1 http://mirror.fcix.net/raspbian/raspbian bookworm/main armhf midori armhf 7.0-2.1 [705 kB]
Fetched 705 kB in 1s (755 kB/s)
Selecting previously unselected package midori.
(Reading database ... 142285 files and directories currently installed.)
Preparing to unpack .../midori_7.0-2.1_armhf.deb ...
Unpacking midori (7.0-2.1) ...
Setting up midori (7.0-2.1) ...
Processing triggers for desktop-file-utils (0.26-1) ...
Processing triggers for hicolor-icon-theme (0.17-2) ...
Processing triggers for gnome-menus (3.36.0-1.1) ...
Processing triggers for libc-bin (2.36-9+rpt2+deb12u10) ...
Processing triggers for man-db (2.11.2-2) ...
Processing triggers for mailcap (3.70+nmul) ...
n1spw@pat-winlink:~ $
```

Figure 13: Install Midori

sudo apt install ax25-tools -y

```
File Edit Tabs Help
n1spw@pat-winlink:~ $ sudo apt install ax25-tools
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  talkd ax25-apps
The following NEW packages will be installed:
  ax25-tools
0 upgraded, 1 newly installed, 0 to remove and 29 not upgraded.
Need to get 169 kB of archives.
After this operation, 2,704 kB of additional disk space will be used.
Get:1 http://raspbian.raspberrypi.com/raspbian bookworm/main armhf ax25-tools armhf 0.0.10-rc5+git20190411+3595f87-6 [169 kB]
Fetched 169 kB in 1s (188 kB/s)
Selecting previously unselected package ax25-tools.
(Reading database ... 142419 files and directories currently installed.)
Preparing to unpack .../ax25-tools_0.0.10-rc5+git20190411+3595f87-6_armhf.deb ...
Unpacking ax25-tools (0.0.10-rc5+git20190411+3595f87-6) ...
Setting up ax25-tools (0.0.10-rc5+git20190411+3595f87-6) ...
Processing triggers for man-db (2.11.2-2) ...
n1spw@pat-winlink:~ $
```

Figure 14: Install ax25-tools

Now that the dependencies are installed, let's install Direwolf.

Run the below commands from a terminal window.

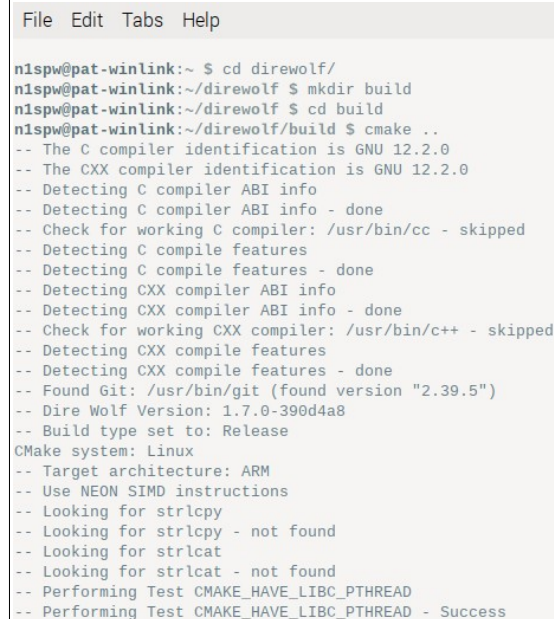
```
$ cd ~/
$ git clone https://www.github.com/wb2osz/direwolf
```



```
File Edit Tabs Help
n1spw@pat-winlink:~ $ git clone https://www.github.com/wb2osz/direwolf
Cloning into 'direwolf'...
warning: redirecting to https://github.com/wb2osz/direwolf.git/
remote: Enumerating objects: 4220, done.
remote: Counting objects: 100% (1357/1357), done.
remote: Compressing objects: 100% (269/269), done.
remote: Total 4220 (delta 1209), reused 1101 (delta 1087), pack-reused 2863 (from 4)
Receiving objects: 100% (4220/4220), 140.38 MiB | 4.11 MiB/s, done.
Resolving deltas: 100% (2889/2889), done.
n1spw@pat-winlink:~ $
```

Figure 15: git clone direwolf

```
$ cd direwolf
$ mkdir build
$ cd build
$ cmake ..
```



```
File Edit Tabs Help
n1spw@pat-winlink:~ $ cd direwolf/
n1spw@pat-winlink:~/direwolf $ mkdir build
n1spw@pat-winlink:~/direwolf $ cd build
n1spw@pat-winlink:~/direwolf/build $ cmake ..
-- The C compiler identification is GNU 12.2.0
-- The CXX compiler identification is GNU 12.2.0
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working C compiler: /usr/bin/cc - skipped
-- Detecting C compile features
-- Detecting C compile features - done
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Check for working CXX compiler: /usr/bin/c++ - skipped
-- Detecting CXX compile features
-- Detecting CXX compile features - done
-- Found Git: /usr/bin/git (found version "2.39.5")
-- Dire Wolf Version: 1.7.0-390d4a8
-- Build type set to: Release
CMake system: Linux
-- Target architecture: ARM
-- Use NEON SIMD instructions
-- Looking for strlcpy
-- Looking for strlcpy - not found
-- Looking for strlcat
-- Looking for strlcat - not found
-- Performing Test CMAKE_HAVE_LIBC_PTHREAD
-- Performing Test CMAKE_HAVE_LIBC_PTHREAD - Success
```

Figure 16: cmake

```
$ make -j4
```

Note: This command will take a while to complete.

```

File Edit Tabs Help
n1spw@pat-winlink:~/direwolf/build $ make -j4
[ 2%] Building C object external/misc/CMakeFiles/misc.dir/strncpy.c.o
[ 2%] Building C object external/geotranz/CMakeFiles/geotranz.dir/error_string.c.o
[ 2%] Building C object external/misc/CMakeFiles/misc.dir/strcat.c.o
[ 2%] Building C object external/geotranz/CMakeFiles/geotranz.dir/mgrs.c.o
[ 2%] Building C object external/geotranz/CMakeFiles/geotranz.dir/polarst.c.o
[ 2%] Linking C static library libmisc.a
[ 3%] Building C object external/geotranz/CMakeFiles/geotranz.dir/tranmerc.c.o
[ 3%] Built target misc
[ 3%] Building C object external/geotranz/CMakeFiles/geotranz.dir/ups.c.o
[ 4%] Building C object external/geotranz/CMakeFiles/geotranz.dir/usng.c.o
[ 4%] Building C object external/geotranz/CMakeFiles/geotranz.dir/utm.c.o
[ 4%] Building C object src/CMakeFiles/decode_aprs.dir/decode_aprs.c.o
[ 5%] Building C object src/CMakeFiles/text2tt.dir/tt_text.c.o
[ 5%] Building C object src/CMakeFiles/tt2text.dir/tt_text.c.o
[ 6%] Linking C static library libgeotranz.a
[ 6%] Built target geotranz
[ 7%] Building C object src/CMakeFiles/log2gpx.dir/log2gpx.c.o
[ 7%] Linking C executable text2tt
[ 7%] Building C object src/CMakeFiles/log2gpx.dir/textcolor.c.o
[ 8%] Linking C executable tt2text
[ 8%] Built target text2tt
[ 9%] Building C object src/CMakeFiles/decode_aprs.dir/ais.c.o
[ 9%] Built target tt2text
[10%] Linking C executable log2gpx
[11%] Building C object src/CMakeFiles/gen_packets.dir/gen_packets.c.o

```

Figure 17: make

\$ sudo make install

```

File Edit Tabs Help
n1spw@pat-winlink:~/direwolf/build $ sudo make install
[ 4%] Built target geotranz
[ 5%] Built target misc
[40%] Built target direwolf
[48%] Built target decode_aprs
[49%] Built target text2tt
[50%] Built target tt2text
[51%] Built target ll2utm
[53%] Built target utm2ll
[55%] Built target log2gpx
[66%] Built target gen_packets
[83%] Built target atest
[85%] Built target aclients
[89%] Built target kissutil
[92%] Built target tnctest
[94%] Built target cm108
[96%] Built target ttcalc
[100%] Built target appserver
Install the project...
-- Install configuration: "Release"
-- Installing: /usr/local/share/doc/direwolf/CHANGES.md
-- Installing: /usr/local/share/doc/direwolf/LICENSE
-- Installing: /usr/local/share/doc/direwolf/external/LICENSE
-- Installing: /usr/local/share/applications/direwolf.desktop
-- Installing: /usr/local/share/pixmaps/direwolf_icon.png
-- Installing: /usr/local/share/direwolf/tocalls.txt
-- Installing: /usr/local/share/direwolf/symbols-new.txt
-- Installing: /usr/local/share/direwolf/symbolsX.txt
-- Installing: /usr/local/bin/direwolf

```

Figure 18: make install

\$ make install-conf

```

File Edit Tabs Help
n1spw@pat-winlink:~/direwolf/build $ make install-conf
Built target install-conf
n1spw@pat-winlink:~/direwolf/build $

```

Figure 19: make install-conf

The last software component is PAT-Winlink.

Run the below commands from a terminal window.

```
$ cd ~/Downloads
```

```
$ wget https://github.com/la5nta/pat/releases/download/v0.16.0/pat_0.16.0_linux_armhf.deb
```

```
File Edit Tabs Help
n1spw@pat-winlink:~ $ cd ~/Downloads
n1spw@pat-winlink:~/Downloads $ wget https://github.com/la5nta/pat/releases/download/v0.16.0/pat_0.16.0_linux_armhf.deb
--2025-05-30 14:17:04-- https://github.com/la5nta/pat/releases/download/v0.16.0/pat_0.16.0_linux_armhf.deb
Resolving github.com (github.com)... 140.82.113.3
Connecting to github.com (github.com)|140.82.113.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/52305585/111aaa26-20e5-4590-9ecd-54fa50cb49c0?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20250530%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20250530T211705Z&X-Amz-Expires=300&X-Amz-Signature=8eb19f8c0baca78fc7d9ef7ce7b0aacdc58608952191dd7fa0c44a5c4580802b&X-Amz-SignedHeaders=host&response-content-disposition=attachment%3B%20filename%3Dpat_0.16.0_linux_armhf.deb&response-content-type=application%2Foctet-stream [following]
--2025-05-30 14:17:05-- https://objects.githubusercontent.com/github-production-release-asset-2e65be/52305585/111aaa26-20e5-4590-9ecd-54fa50cb49c0?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetproduction%2F20250530%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20250530T211705Z&X-Amz-Expires=300&X-Amz-Signature=8eb19f8c0baca78fc7d9ef7ce7b0aacdc58608952191dd7fa0c44a5c4580802b&X-Amz-SignedHeaders=host&response-content-disposition=attachment%3B%20filename%3Dpat_0.16.0_linux_armhf.deb&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.110.133, 185.199.111.133, 185.199.109.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.110.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3244958 (3.1M) [application/octet-stream]
Saving to: 'pat_0.16.0_linux_armhf.deb.2'

pat_0.16.0_linux_armhf.deb.2 100%[=====] 3.09M 5.50MB/s in 0.6s

2025-05-30 14:17:06 (5.50 MB/s) - 'pat_0.16.0_linux_armhf.deb.2' saved [3244958/3244958]

n1spw@pat-winlink:~/Downloads $
```

Figure 20: wget PAT-Winlink

```
$ sudo dpkg -i pat_0.16.0_linux_armhf.deb
```

```
File Edit Tabs Help
n1spw@pat-winlink:~/Downloads $ sudo dpkg -i pat_0.16.0_linux_armhf.deb
(Reading database ... 142399 files and directories currently installed.)
Preparing to unpack pat_0.16.0_linux_armhf.deb ...
Unpacking pat (0.16.0) over (0.16.0) ...
Setting up pat (0.16.0) ...
Processing triggers for man-db (2.11.2-2) ...
n1spw@pat-winlink:~/Downloads $
```

Figure 21: Install PAT-Winlink package

All the required software is now installed on the Pi. Next step is to do some configuration.

Step-3: Configuration

Direwolf

The Direwolf configuration file is named *direwolf.conf*. You will find it in your home (~/) folder.

There are four (4) things Direwolf needs from you in order to operate correctly.

- The device ID of your Digirig
- Your call sign
- The baud rate of your packet communication
- The correct PTT setting

Before we change the Direwolf configuration file, let's determine what USB device is assigned to your Digirig. Plug your Digirig USB cable into the Digirig USB-C port and the other into a USB port on your Pi.

In a terminal window, type the below command.

```
$ arecord -l
```



```
File Edit Tabs Help
n1spw@pat-winlink:~/Downloads $ arecord -l
**** List of CAPTURE Hardware Devices ****
card 2: Device [USB PnP Sound Device], device 0: USB Audio [USB Audio]
Subdevices: 1/1
Subdevice #0: subdevice #0
n1spw@pat-winlink:~/Downloads $
```

Figure 22: arecord -l

Your Digirig will show up as “USB PnP Sound Device”. You can see my Digirig device is assigned to card-2, device-0.

From a terminal window, execute the below commands:

```
$ cd ~/
$ nano direwolf.conf
```

This will open the Direwolf configuration file in a text editor.

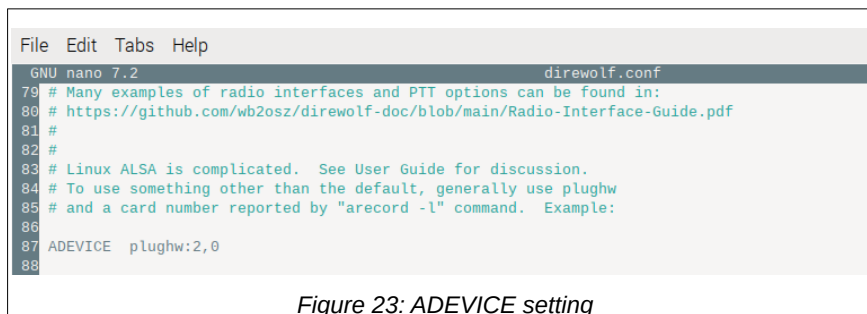
Note: Once nano is running, you can display line numbers with <Alt><Shift><3>.

Scroll down to somewhere around line 87. You are looking for a line contains “# ADEVICE plughw: 1,0”. The # symbol indicates the line is a comment and not a configuration value. We need to change this.

Delete the “#” and space so the line contains: ADEVICE plughw: 1,0.

Next replace the 1,0 with the card and device number shown by the arecord -l command.

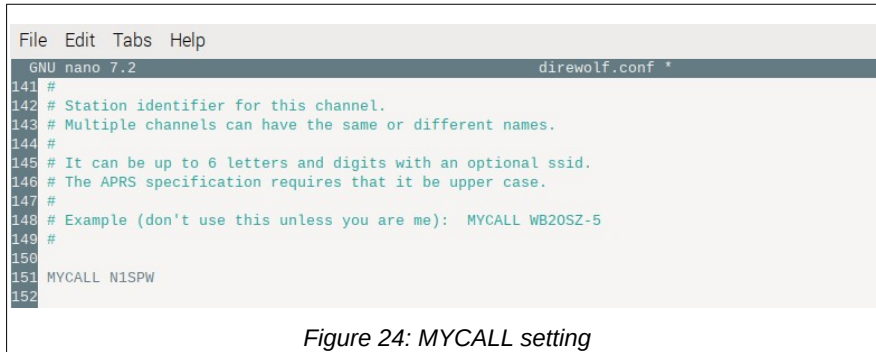
In my case, the entry is: ADEVICE plughw: 2,0.



```
File Edit Tabs Help
GNU nano 7.2 direwolf.conf
79 # Many examples of radio interfaces and PTT options can be found in:
80 # https://github.com/wb2osz/direwolf-doc/blob/main/Radio-Interface-Guide.pdf
81 #
82 #
83 # Linux ALSA is complicated. See User Guide for discussion.
84 # To use something other than the default, generally use plughw
85 # and a card number reported by "arecord -l" command. Example:
86
87 ADEVICE plughw:2,0
88
```

Figure 23: ADEVICE setting

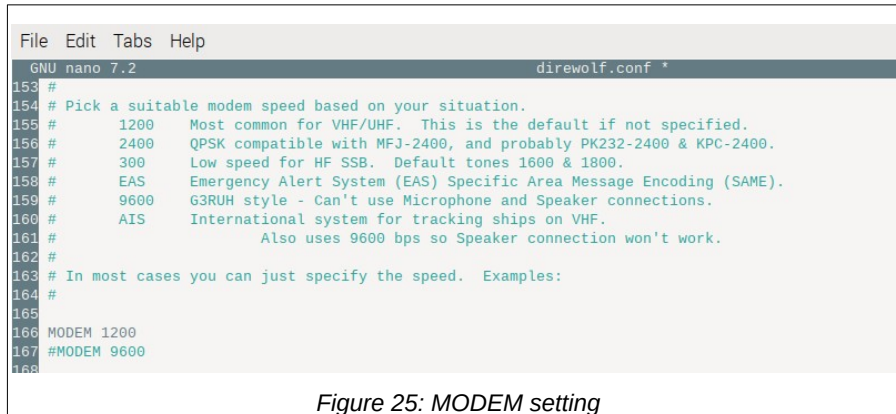
Scroll down to somewhere around line 151.
Replace the NOCALL entry with your call sign.



```
File Edit Tabs Help
GNU nano 7.2 direwolf.conf *
141 #
142 # Station identifier for this channel.
143 # Multiple channels can have the same or different names.
144 #
145 # It can be up to 6 letters and digits with an optional ssid.
146 # The APRS specification requires that it be upper case.
147 #
148 # Example (don't use this unless you are me): MYCALL WB20SZ-5
149 #
150
151 MYCALL N1SPW
152
```

Figure 24: MYCALL setting

Scroll down to somewhere around line 166.
Uncomment the line that contains the Baud rate you will use. In most cases this will be MODEM 1200.



```
File Edit Tabs Help
GNU nano 7.2 direwolf.conf *
153 #
154 # Pick a suitable modem speed based on your situation.
155 # 1200 Most common for VHF/UHF. This is the default if not specified.
156 # 2400 QPSK compatible with MFJ-2400, and probably PK232-2400 & KPC-2400.
157 # 300 Low speed for HF SSB. Default tones 1600 & 1800.
158 # EAS Emergency Alert System (EAS) Specific Area Message Encoding (SAME).
159 # 9600 G3RUH style - Can't use Microphone and Speaker connections.
160 # AIS International system for tracking ships on VHF.
161 # Also uses 9600 bps so Speaker connection won't work.
162 #
163 # In most cases you can just specify the speed. Examples:
164 #
165
166 MODEM 1200
167 #MODEM 9600
168
```

Figure 25: MODEM setting

Finally, we need to determine how to activate the PTT (Push-to-talk) on your radio. The Digirig provides two interfaces. One is audio, and the other is serial. We already configured the audio with the ADEVICE setting.

To determine what serial device to use, we need to identify it. To do this we need to list the Pi USB devices. Execute the below command in a terminal window.

```
$ ls -l /dev/USB*
```

This command will show the active USB devices. In my case, the only USB device in use is /dev/ttyUSB0.



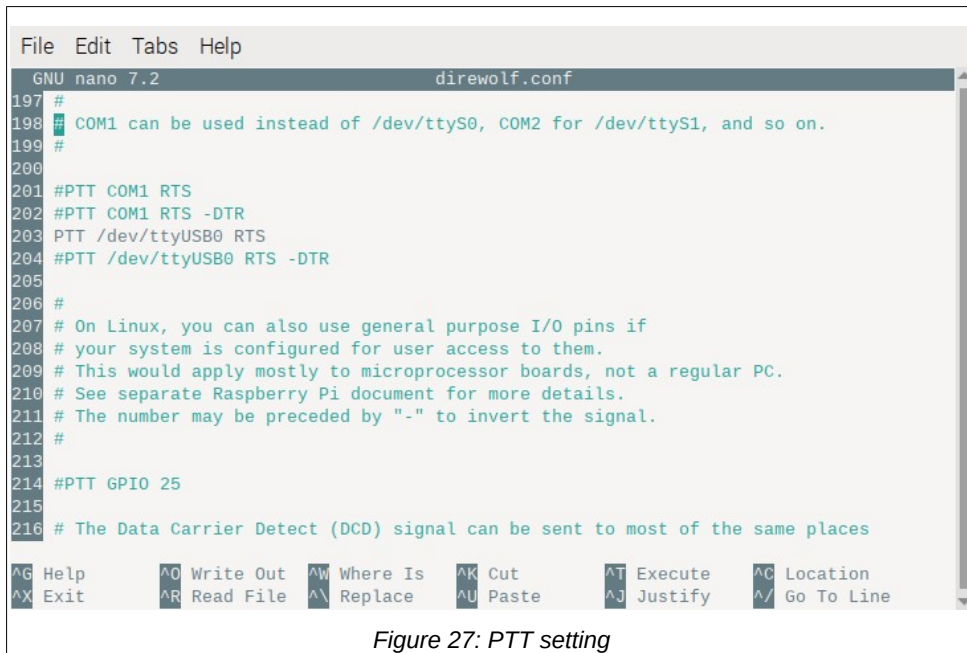
```
File Edit Tabs Help
n1spw@pat-winlink:~ $ \ls -l /dev/ttyUSB*
crw-rw---- 1 root dialout 188, 0 May 31 18:07 /dev/ttyUSB0
n1spw@pat-winlink:~ $
```

Figure 26: List USB devices

If you see more than one USB device in use, unplug your Digirig. The USB device it was assigned will disappear when you re-run `$ ls -l /dev/USB*`.

In the Direwolf config file, scroll down to around line 200. Here you will see the PTT settings. For most radios you will configure PTT using the RTS setting. As shown below, on line 203 it shows the PTT setting should

use /dev/ttyUSB0 – the device identified above, and enable RTS.



```
File Edit Tabs Help
GNU nano 7.2 direwolf.conf
197 #
198 # COM1 can be used instead of /dev/ttyS0, COM2 for /dev/ttyS1, and so on.
199 #
200
201 #PTT COM1 RTS
202 #PTT COM1 RTS -DTR
203 PTT /dev/ttyUSB0 RTS
204 #PTT /dev/ttyUSB0 RTS -DTR
205
206 #
207 # On Linux, you can also use general purpose I/O pins if
208 # your system is configured for user access to them.
209 # This would apply mostly to microprocessor boards, not a regular PC.
210 # See separate Raspberry Pi document for more details.
211 # The number may be preceded by "-" to invert the signal.
212 #
213
214 #PTT GPIO 25
215
216 # The Data Carrier Detect (DCD) signal can be sent to most of the same places
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

Figure 27: PTT setting

Save the Direwolf config file - <Ctrl><s> <Ctrl><x>.

PAT-Winlink

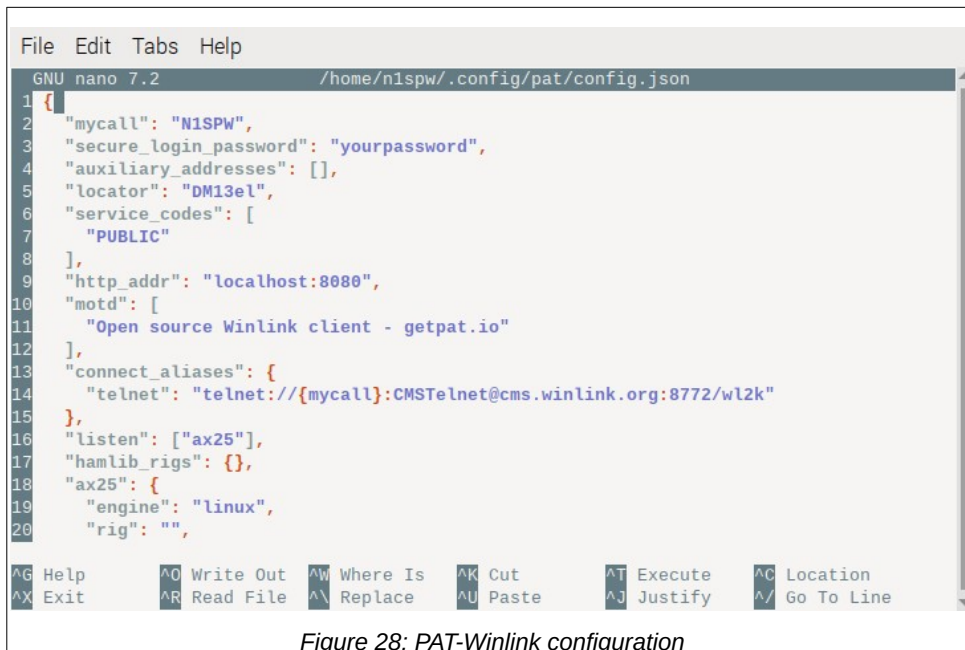
To configure PAT-Winlink, from a terminal window enter the command:

`$ pat configure.`

PAT-Winlink uses a JSON configuration file. It is important that you get the file syntax correct. Pay attention to the braces, quotes, and commas. If one is missing or in the wrong place, the entire configuration file will become invalid.

There are 2 things PAT-Winlink needs from you in order to operate correctly.

- Your call sign, Winlink password, and gridsquare.
- Enable listen mode so you can receive incoming P2P messages.



```
File Edit Tabs Help
GNU nano 7.2 /home/n1spw/.config/pat/config.json
1 {
2   "mycall": "N1SPW",
3   "secure_login_password": "yourpassword",
4   "auxiliary_addresses": [],
5   "locator": "DM13el",
6   "service_codes": [
7     "PUBLIC"
8   ],
9   "http_addr": "localhost:8080",
10  "motd": [
11    "Open source Winlink client - getpat.io"
12  ],
13  "connect_aliases": {
14    "telnet": "telnet://{mycall}:CMSTelnet@cms.winlink.org:8772/wl2k"
15  },
16  "listen": ["ax25"],
17  "hamlib_rigs": {},
18  "ax25": {
19    "engine": "linux",
20    "rig": ""

```

Figure 28: PAT-Winlink configuration

At the top of the config file enter the following:

- Your call sign in quotes next to "mycall": .
- Your Winlink password in quotes next to: "secure_login_password":.
- Your gridsquare next to "locator":
- Insert "ax25" inside the empty braces next to "listen:"

Use the above screenshot as a guide for your entries. Once you are sure your syntax is correct, save the configuration file with <Ctrl><s> <Ctrl><x>.

AX.25 Port

To configure AX.25 we need to add a line to a configuration file.

Enter the below command:

```
$ sudo nano /etc/ax25/axports
```

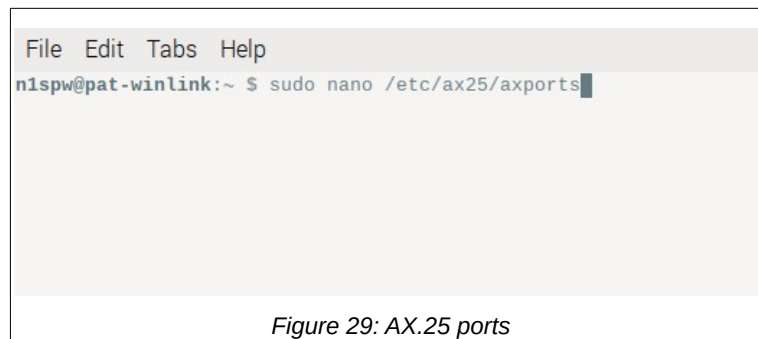


Figure 29: AX.25 ports

Add a line to the bottom of the file as shown highlighted below:

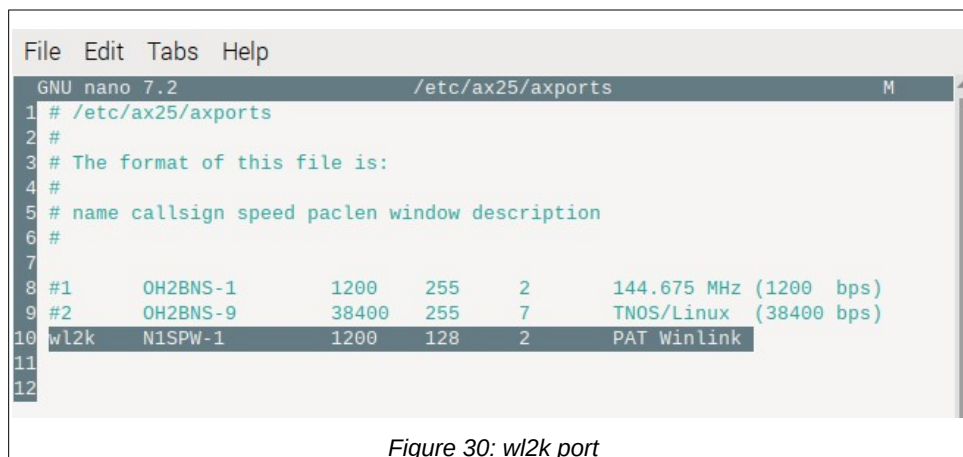


Figure 30: wl2k port

Enter the values as shown. Change the call sign to your call sign followed by a -1. Be sure to separate the fields with tabs. Save the file <Ctrl> <s> <Ctrl> <x>.

Step-4: Radio/Comms

Now that the required software is installed and configured, it is time to get on the air and send some Emails.

In this step we will do the following:

1. Configure your radio for Winlink packet mode.
2. Start Direwolf and ensure it connects to your Digirig.
3. Set proper volume levels.
4. Start PAT-Winlink and send an Email via the Internet.
5. Send a Winlink Email via radio to a local RMS.

1. Configure radio for Winlink packet mode

In this "How-To" I am using a Baofeng UV-5R. I have also successfully sent Winlink messages using an Anytone D878UV Plus, Yaesu FT-60, VX-6R, FT-300D mobile and a Kenwood TM-D700 mobile. In Appendix II, you will find the radio settings I use for each of these radios.

Refer to the Appendix II and configure your radio.

2. Start Direwolf and ensure it connects to your Digirig

Direwolf is a terminal application. There is no GUI. This is why I like it. It simply runs in a terminal window and tells you exactly what it is seeing and doing.

On your Pi, open a terminal window and enter the below command:

```
$ sudo direwolf -p
```

If Direwolf can find your Digirig, you will see a message similar to Figure 31.

```
Audio device for both receive and transmit: plughw:2,0 (channel 0)
Channel 0: 1200 baud, AFSK 1200 & 2200 Hz, A+, 44100 sample rate / 3.
Ready to accept AGW client application 0 on port 8000 ...
Ready to accept KISS TCP client application 0 on port 8001 ...
DNS-SD: Avahi: Announcing KISS TCP on port 8001 as 'Dire Wolf on pat-winlink'
Virtual KISS TNC is available on /dev/pts/3
Created symlink /tmp/kisstnc -> /dev/pts/3
DNS-SD: Avahi: Service 'Dire Wolf on pat-winlink' successfully registered.
```

Figure 31: Direwolf console

If Direwolf cannot find your Digirig, make sure your Direwolf config file has the correct device configured and that the Digirig cables are plugged in tightly. This is especially important for Baofeng radios. The dual-prong Digirig cable must be pushed in securely.

3. Set proper volume levels

It is very important the proper audio volume levels are set correctly. This is not difficult, but it can be a little tedious. From a terminal window enter the below command:

```
$ alsamixer
```

This will start the Alsa Mixer terminal app. To select your Digirig device, press the F6 key. Scroll down using the arrow keys until the "USB PnP Sound Device" is highlighted. Press <Enter>.

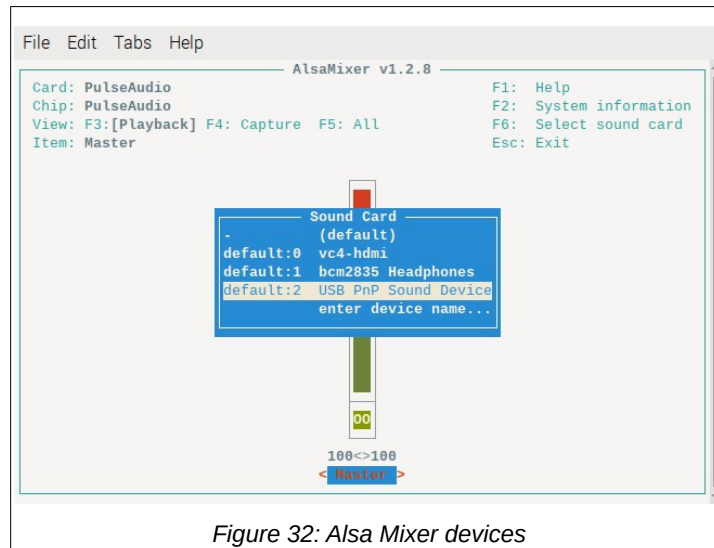


Figure 32: Alsa Mixer devices

Now you will see three things you can configure: Speakers, Microphone, and Gain control. To move between the three settings, use the right/left arrow keys. To adjust a volume level, use the up/down arrow keys.

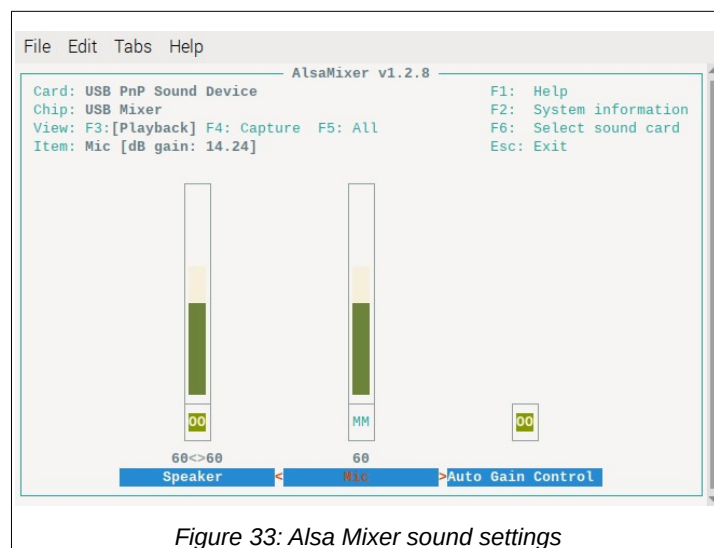


Figure 33: Alsa Mixer sound settings

Adjust the speaker volume to @60. Use the right arrow key to highlight the microphone setting. Set the microphone setting to @60. Make sure the Auto Gain Control (MM) setting is set to 0. Once you have your settings correct, press the <Esc> key.

NOTE: Direwolf complains a lot about volume levels. Most of the time it will tell you your volume levels are too high. I do not pay much attention to these warnings. If you cannot connect to an RMS due to incorrect volume levels, this usually means your settings are too high. Start @60 and experiment.

4. Start PAT-Winlink and send Email via the Internet

Next, we will start PAT-Winlink and send an Email via the Internet. PAT-Winlink requires two commands to run.

First, we must point PAT-Winlink to Direwolf using the below commands:

```
$ LINK=$(ls -l /tmp/kisstnc | awk '{print $NF}')
```

```
$ sudo kissattach $LINK wl2k
```

```
$ sudo kissparms -c 1 -p wl2k
```



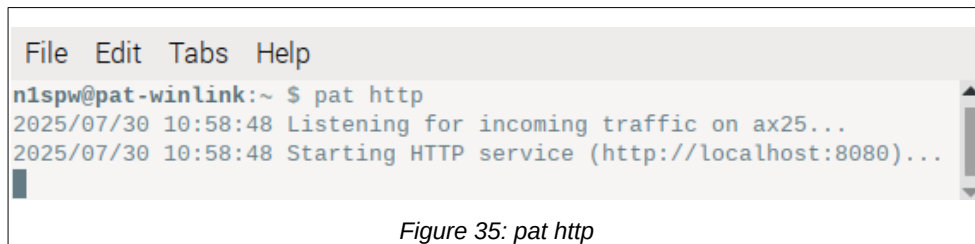
```
File Edit Tabs Help
n1spw@pat-winlink:~ $ sudo kissattach /tmp/kisstnc wl2k
AX.25 port wl2k bound to device ax0
n1spw@pat-winlink:~ $
```

Figure 34: Kissatach

Next, from start PAT-Winlink:

```
$ pat http
```

The http argument tells PAT-Winlink to start the web service.



```
File Edit Tabs Help
n1spw@pat-winlink:~ $ pat http
2025/07/30 10:58:48 Listening for incoming traffic on ax25...
2025/07/30 10:58:48 Starting HTTP service (http://localhost:8080)...
```

Figure 35: pat http

Open a web browser and go to the URL: <https://localhost:8080>.

NOTE: Earlier, you installed a lightweight browser named Midori. I recommend you use it. If everything worked, you should see the PAT-Winlink web application as shown below.

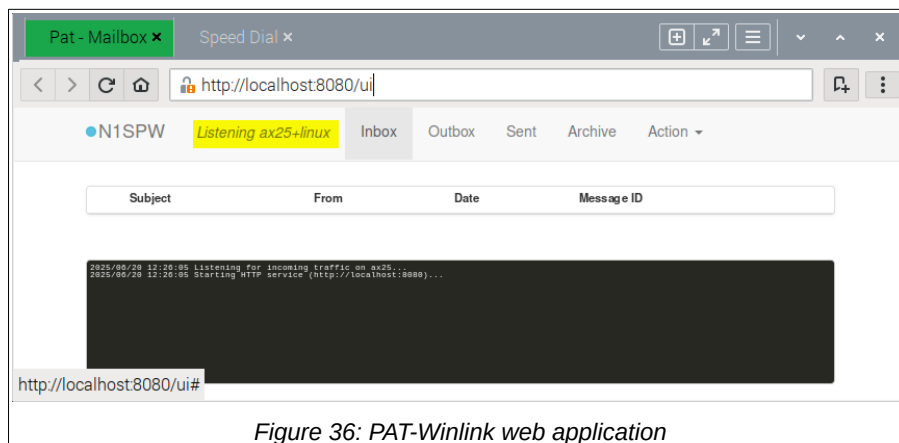


Figure 36: PAT-Winlink web application

Notice the “Listening ax25+linux”. This means PAT-Winlink is listening for P2P connections.

For the next step, be sure your Raspberry Pi is connected to the Internet. Click on the ‘Action’ dropdown.

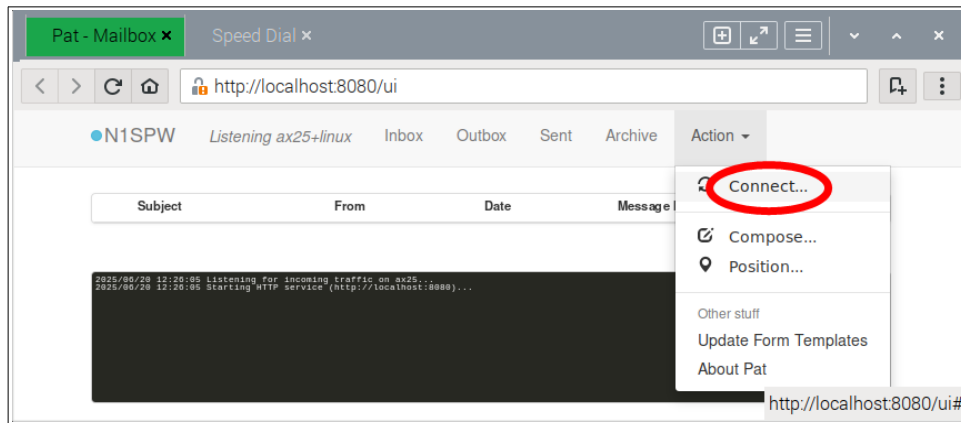


Figure 37: Compose Email

Click on "Connect"

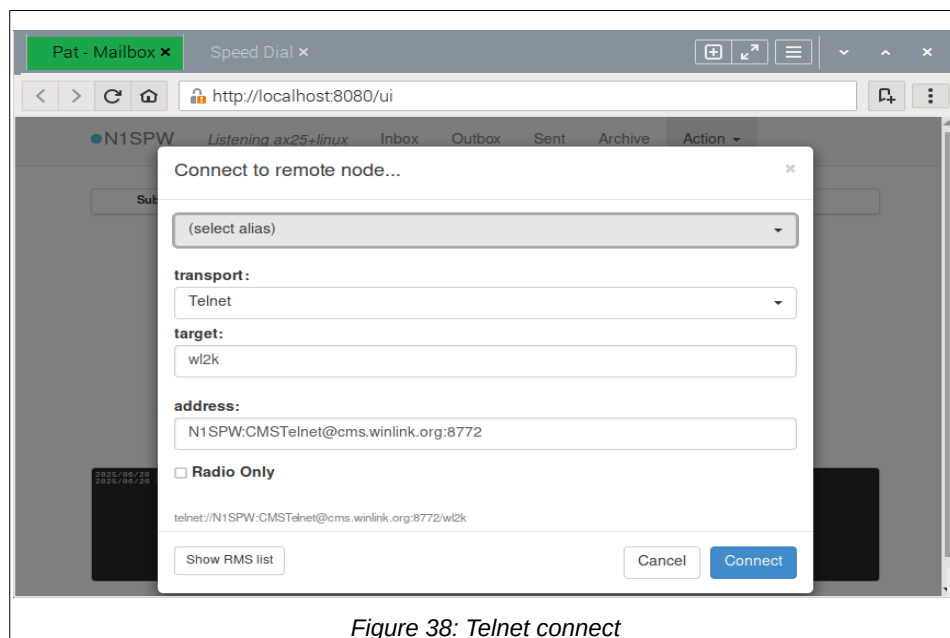


Figure 38: Telnet connect

From the (select alias) dropdown – select 'Telnet'.

The dialog will be populated with your call sign and connect information in the 'address' box.

Click the 'Connect' button.

It may be hard to see due to the small font, but in the black console box you should see PAT-Winlink connect to the Internet and send/receive your Emails. In my case, I had no pending in/out Emails, so the application simply disconnected.

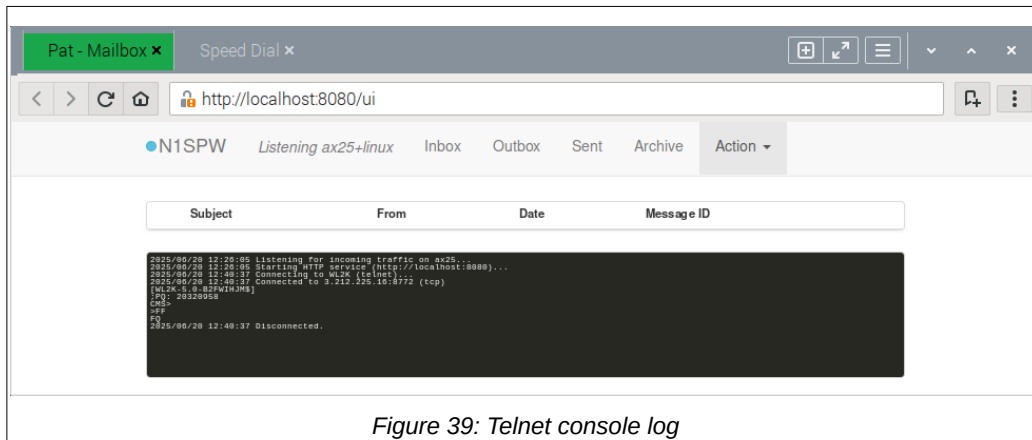


Figure 39: Telnet console log

If your PAT-Winlink does not connect to the Internet and/or gives you errors, refer the the Troubleshooting guide in Appendix I.

The below screenshot shows my Desktop when Direwolf and PAT-Winlink are running.

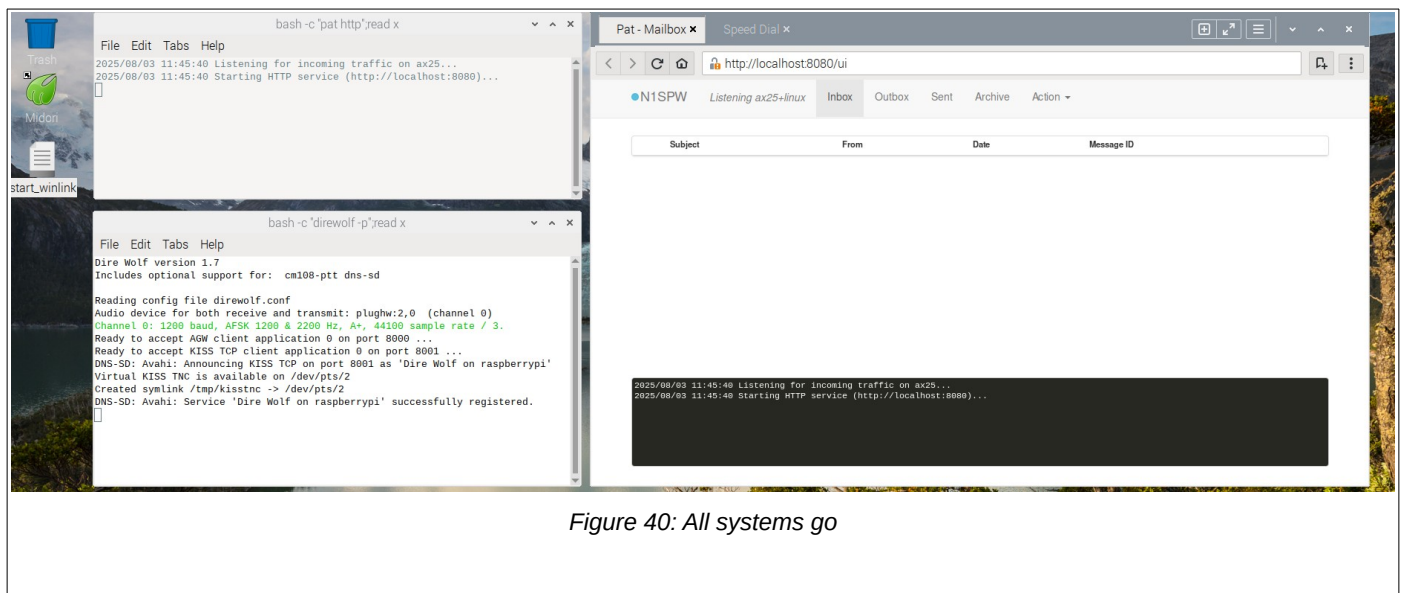


Figure 40: All systems go

The PAT-Winlink and Direwolf console windows will tell you everything they are doing. This is great for troubleshooting and seeing what is going on in the system.

5. Send a Winlink Email via radio to a local RMS

You are now be ready to send a Winlink Email via your radio. For this step to be successful, you must be close enough to a Winlink RMS for your radio to connect. Remember, a typical H/T only transmits @5 watts, so the RMS must be pretty close.

Create a test Email message and make sure it is sitting in your Outbox. Click on “Action” | “Compose”. Enter an Email address, subject and body content. Then click ‘Post’.

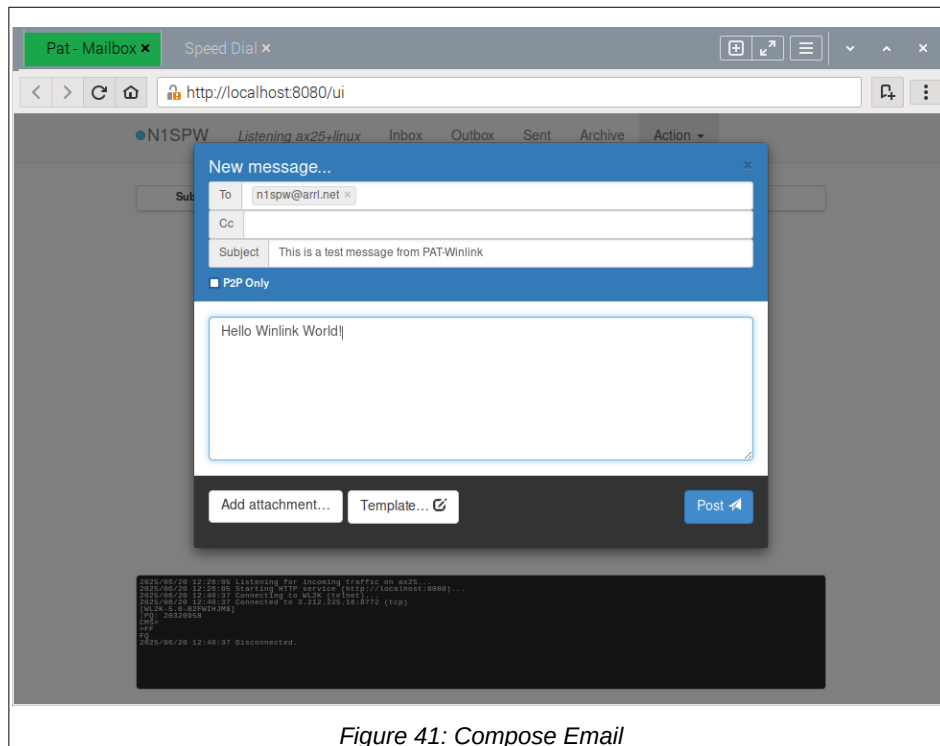


Figure 41: Compose Email

You should now see the message in your ‘Outbox’. Click on ‘Outbox’ to see it.

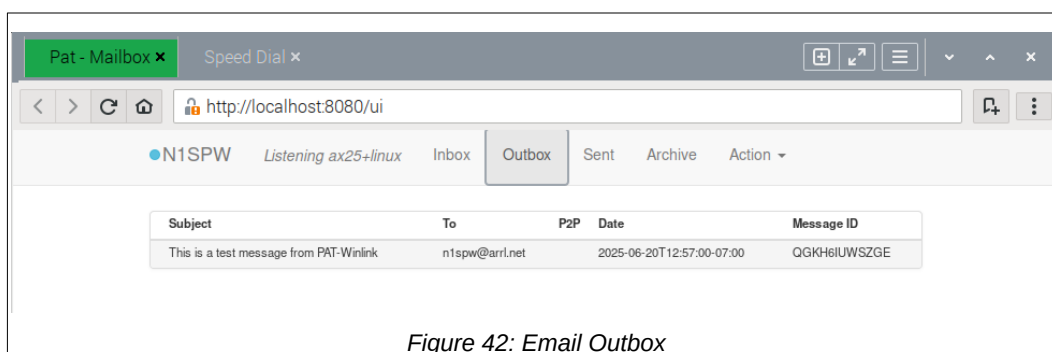
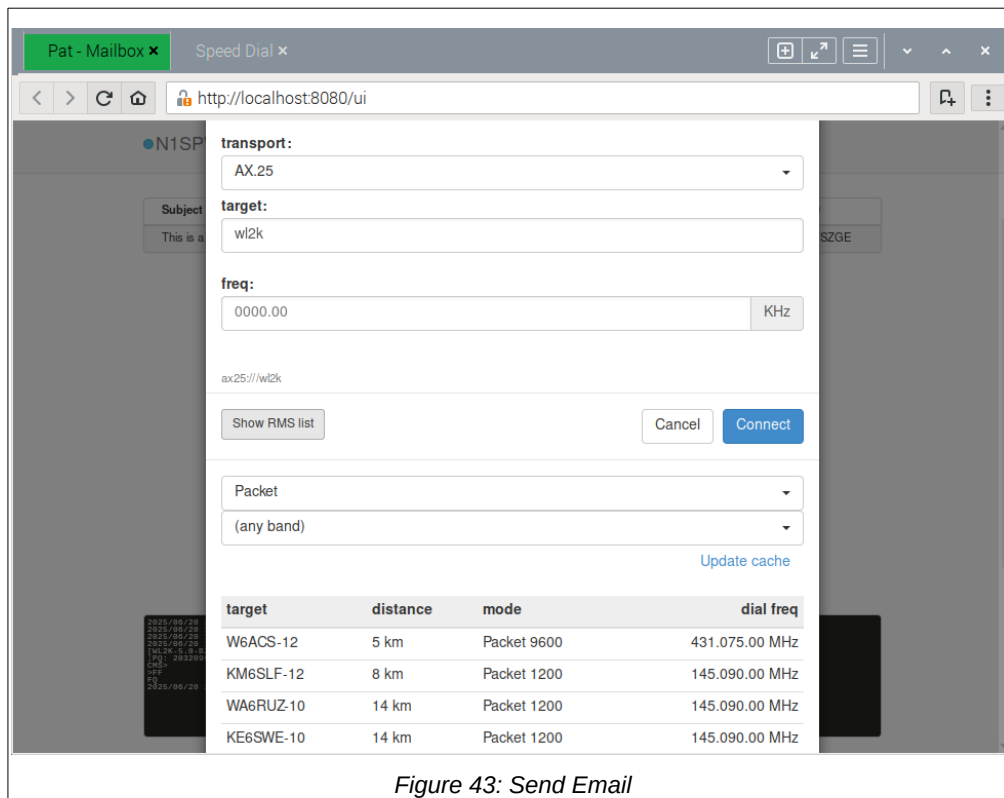


Figure 42: Email Outbox

Make sure your radio is properly configured, and it is securely connected to your Digirig. Turn the squelch on your radio to 0 (open).

From PAT-Winlink click on ‘Action’ | ‘Connect’



Next, set your *transport* mode to AX.25. The *target* will automatically be set to wl2k. Leave the frequency blank. Click the 'Show RMS list' button. Select "Packet". Choose an RMS that is close enough for you to connect. Once selected, be sure to tune your radio to the same frequency.

NOTE: It is handy to have another H/T set to the same frequency so you can hear the connection to the RMS.

Click 'Connect'

When your radio attempts to connect to the RMS, you can see what is happening in the lower PAT-Winlink console window and the Direwolf terminal window. Both of these are very useful when troubleshooting.

You should see your radio connect to the RMS, and a series of messages sent back and forth. When your message send is complete, the RMS will disconnect. You will then notice that your Email has been moved from your Outbox folder to your Sent folder.

Congratulations. Your hard work has paid off. You now have a working PAT-Winlink setup that you can use in the field.

Step-5: Operations

You need to ensure you can quickly get your Winlink station up and running in the future. For this, I like to use a checklist. Below are the things you must do to get your station up and running after the system is shutdown.

- ✓ Connect the Digirig to your radio
- ✓ Connect the Digirig to your Pi
- ✓ Boot the Pi (I run my Pi without a monitor and connect to it via VNC)
- ✓ Turn on the radio and open the squelch (0)
- ✓ Configure the correct sound levels (`$ alsamixer`)
 F6 | Select device: USB PnP Sound device
 Speaker @60, Microphone @60, Auto Gain Control @00
NOTE: Use tab and arrow keys to traverse, <Esc> to exit
- ✓ Start Direwolf from a terminal window
`$ sudo direwolf -p`
- ✓ Start kissattach from a terminal window
`$ LINK=$(ls -l /tmp/kisstnc | awk '{print $NF}')`
`$ sudo kissattach $LINK w12k`
`$ sudo kissparms -c 1 -p w12k`
- ✓ Start PAT Winlink from a terminal window
`$ pat http`
- ✓ Start Midori browser and enter URL: `http://localhost:8080`
- ✓ If Direwolf does not start. Ensure `direwolf.conf` file has the correct Digirig device Id, Open a terminal window and determine the Digirig device Id (`$ arecord -l`). Make sure it matches the Direwolf ADEVICE setting in `direwolf.conf`.
`ADEVICE plughw: 2,0`
- ✓ If Direwolf does not start. Ensure `direwolf.conf` file has the correct USB port setting. Open a terminal window and determine the Digirig USB port (`$ ls /dev/ttyUSB*`)
 e.g - `/dev/ttyUSB0`
- ✓ If PTT does not work, ensure `direwolf.conf` file has the correct PTT setting for your radio
 e.g. `PTT /dev/ttyUSB0 RTS`

There are quite a few steps to get up and running. Once you do it a few times, it goes very fast. Here are a couple of suggestions:

- Plug the Digirig into the same USB port each time. In most cases, the Pi will assign the same device Id.
- You will need to adjust the sound levels for each radio you use. I ignore the Direwolf audio level warnings, and adjust the sound levels that work with each radio.
- Make sure the volume on your radio is correct. Set it to 50-60%. Correct audio settings on the computer and the radio must be at the correct levels or you will not connect to an RMS.

NOTE: I have created a bash script that starts Direwolf and PAT Winlink for you. It is included in the download package with this document. I highly recommend you use it. You can learn more about this script in Appendix IV.

The reason I like to use the Raspberry Pi for field Winlink use is its small size and low power consumption. Also remember, once configured correctly, you can use your Pi SD Card in any late model Pi. You can distribute multiple Pi SD Cards for other Winlink operators to use. This is really handy.

Wrap-Up

I hope you find this guide useful and that it helps you get a PAT Winlink station up and running.

My intent here is to provide some detailed, visual, documentation, to help new Hams or those do have little patience for the complications of technology. I also hope you learned something.

Please send corrections, comments, complaints, ideas, or any other feedback to: n1spw@tuta.com.

73,
N1SPW
Aug 9, 2025

Appendix I: Troubleshooting

Sending an Email via radio waves is a complex process. As you can see from the previous pages, there are a lot of parts that need to work together to get an Email from a radio to a recipient inbox. A lot can go wrong.

Depending on the issue you are having, the Internet support forums are a great help. Winlink and *Digirig* have very active forums that have experienced Hams ready to give you advice. Denis, the creator of the Digirig is very active in his support forum and has been extremely generous with his time and genius to help anyone who asks.

Based on my experience and digging around support forums, here are the areas that cause the trouble:

1. Direwolf configuration settings
2. PAT Winlink configuration settings.
3. Push-To-Talk (PTT) setting
4. Volume controls
5. Radio comms
6. Connecting to a Winlink RMS

Direwolf configuration settings

If you cannot get Direwolf to start, it usually tells you what is wrong. In most cases, it cannot find the Digirig device. Make sure the ADEVICE setting in the direwolf.conf file matches the Digirig device Id:
(*\$ arecord -l*).

Also, make sure you have a USB cable that works with the Digirig. I cannot tell you many times I have read in a forum the reason the Digirig did not work was because of an improper USB cable. Some USB cables are only wired for charging devices. These cables will not work with a Digirig. When in doubt, try a different USB cable.

PAT Winlink configuration settings

Configuring PAT Winlink is straightforward. You run *\$ pat configure* from a terminal window and change a few settings. The danger here is the PAT configuration file is in JSON format. This means that one misplaced comma, colon, or brace "*breaks*" the entire file. You will know this has happened when PAT Winlink is unable to load its configuration file.

If your PAT Winlink configuration file is completely broken, you can create a new one. To do so, you will have to remove the broken file. Enter the below commands in a terminal window:

```
$ cd ~/
$ cd .config/pat (Notice the period before the word config)
$ rm config.json
$ pat configure (This command will generate a new config file and open an editor.)
```

Push-To-Talk (PTT)

Getting the *Digirig* to send a serial PTT signal to your radio can be a challenge. I suspect this is the most common issue Hams have with the *Digirig*. Denis (*Digirig* creator) has done a great service building customized cables for so many radios. He also freely provides cable schematics, and encourages Hams to build their own cables.

I can only provide general advice here. The key is to ensure that the Direwolf PTT setting matches the USB port assigned to the Digirig. If you have trouble with PTT, sometimes it helps to unplug, then reconnect the Digirig USB cable from the computer. A reboot can also clear up problems if you have been making a lot of settings changes.

Also be sure the radio cable is securely seated in the radio. This is a particular problem for the Yaesu Vx-6R cable with a threaded tip.

Free Advice: If you are using a Vx-6R, I suggest you remove the o-ring from the Digirig cable. This will ensure the tip seats completely in the radio.

On many radios, you must enable/disable menu items to get PTT to work. In particular, you may need to ensure VOX is disabled.

If PTT is a problem for you, the Digirig support forum is the place to find detailed troubleshooting information for your particular radio.

Volume Control

Using alsamixer to set the sound volume is a little clunky, but it works. Make sure you have right sound device selected [USB PnP Sound Device], and start with the speaker and microphone settings @60. Also make sure the Auto Gain Control setting is off [00].

Direwolf complains a lot about sound levels being too high. I ignore these warnings and move the sound values up and down until it works. I error on the sound settings being too high, than too low.

Also remember, you need to have the squelch on your radio set to zero.

Radio settings

You should not have to make a lot of changes to your radio setting to get all this to work. The way I determined what settings I needed for each of my radios was to reset them to factory default settings. Then I made as few changes as I needed to the default settings to get the rig to work.

You may need to experiment to find the right formula. One area to pay attention to is VOX and squelch tail elimination. The other is the radio volume level.

The key to radio settings is to ensure your audio volume levels are correct, and that you xmit a clear signal with a clean/fast break at the end. The xmit/rcv cycle between your radio and the RMS is near instantaneous. There can be no squelch tail, beeps, or anything else preventing your radio from instant rcv after a transmit.

One other thing that can catch you – repeater offsets. Be sure to observe your radio display when transmitting. If you see it switches channels on xmit, you have a repeater offset in play. You need to be sure you are in simplex mode on all frequencies when talking to an RMS.

As a final troubleshooting step, I again want to emphasize, the importance of having an distortion free radio signal and clean xmit cutoff.

In my opinion, if you can get the Digirig PTT circuit working with your radio, it should be able to send Emails on FM.

You can find the configuration setting for the radios I use in Appendix II.

Connecting to a Winlink RMS

Many Winlink users complain they cannot connect to a Winlink RMS even though they can hear their radio sending the packet tones. Everything seems to be working, but the RMS does not “answer”. Although this sounds dumb, are you sure you are close enough to the RMS that it hears you?

If you are trying to send Winlink Emails with an H/T, most H/T transmit at about 5W max. Unfortunately, this means your radio signal is not going to carry very far. Even if you hook your H/T up to a large and capable antenna, you are still only broadcasting @5 watts.

In other words, you need to be really close to the RMS for it to work. If possible, I suggest you get your Winlink station up and running on a 50W mobile radio, before experimenting with H/T's.

Appendix II: Radio Settings

This section contains a general guide on radio settings for Winlink communications. I have successfully sent Winlink messages with the below radios:

- AnyTone D878UV Plus
- Baofeng BF-F8HP
- Baofeng UV5R
- Yaesu FT-60 H/T
- Yaesu FTM-300D Mobile
- Yaesu Vx6R H/T

You may be surprised to learn there are very few setting changes you need to make to get your rig to send Emails via VARA/Winlink. In fact, my Yaesu FT-60, and VX-6R only needed the squelch to be set to 0 (Off) after a factory reset, and they were sending Emails.

To test the radio settings on my radios, I first reset them to factory specs. This ensured that all settings were at their default settings. Then I connected the correct Digirig cable to the radio and made sure it was secure. Finally, I tested each radio without making any changes. If it did not work, then I started to tweak settings.

Below are my recommended radio settings:

- The frequency you choose *must* be in simplex mode.
- The squelch should be turned off (Set to 0).
- PTT signal from your Digirig must work.
- VOX should be off.
- Dual-mode should be off.
- Power-saver mode on both Xmit and Rcv should be off.

NOTE: These settings are only a guide. Your radio may need other setting changes. Experiment until you achieve success.

If you have a radio that is programmed to your liking and you are not interested in a factory reset, I suggest you connect it to the Digirig and see if it works as is. If it does not connect to an RMS, then compare the radios factory default settings and see what is different.

NOTE: During all of the radio tests, I only changed out the Digirig cable for the specific radio I was testing. I did not have to change any of the Direwolf or audio settings on the computer. It just worked.

AnyTone D878UV Plus

To factory reset this radio you press the PTT and PF1 (Just below PTT) switches together and then turn on the radio. The radio will ask you to confirm a reset.

Set the clock time if desired (Menu | Settings | Radio Set | Other Func | 27-Time Zone & 29-Date Time).

Set VFO-A channel to the RMS channel (e.g 145.090)

Set Analog mode (Menu | Settings | 2-Channel Set | 3-Channel Type | 1-A-Analog).

Set Squelch level (Menu | Settings | 1-Radio Set | 4-Other Func | 6-Ana Sq Level | 1-Ana SQ Off).

Without any other changes, I was able to connect to an RMS and send an Email.

Baofeng BF-F8HP

I reset the radio and turned the squelch to 0.

Set Squelch level (Menu | Item-0 | Set to 0 | Menu)

Turn the radio volume to about 75%.

Adjust the Tx power to fit the distance from an RMS.

NOTE: Baofeng radios require the H/T volume setting to be quite high to connect to an RMS.

I was able to connect to an RMS without any issues. The Baofeng radios do not have the highest fidelity, so they typically need higher volume settings to perform Winlink tasks well.

Baofeng UV-5R

To my surprise, this radio sends Winlink messages just fine. To factory reset this radio press Menu, then scroll to Item 40. Press Menu twice. When the radio reboots perform the following:

Set Squelch level (Menu | Item-0 | Set to 0 | Menu)

Turn the radio volume to about 75%

I was able to connect to an RMS without any other changes. Some Internet chats suggest you need to turn squelch-tail-elimination off (Menu Item 35).

Yaesu FT-60

This H/T is one of my favorite radios. I highly recommend this to new Hams as their first radio.

To factory reset the radio, hold down the monitor button just below the PTT, then turn on the radio. Select item number 4 from the list. Press and hold the F/W button until the radio reboots.

When the radio reboots perform the following:

Set Squelch level to 0 (Collar dial on top of radio all the way to the left).

Turn the radio volume to about 60%

Adjust the Tx power to fit the distance from an RMS.

I was able to connect to an RMS without any other changes. Yaesu radios seem to work right out of the box.

Yaesu FTM-300D

This is a mobile rig. There are a lot of settings to this radio, but it connects to an RMS after a factory reset without any configuration changes. There is one thing to note here. This radio will only do digital on VFO B. This means the active VFO must be the bottom one set to the correct frequency for the RMS.

Yaesu Vx6

I own three of these radios. When I go into the field, this is the radio I take with me.

To factory reset the radio, hold down the monitor button just below the PTT, then turn on the radio. Select item number 4 from the list. Press and hold the F/W button until the radio reboots.

When the radio reboots perform the following:

Set Squelch level to 0 (F/W | Monitor button below PTT | Turn collar knob on top of radio to the left).

Turn the radio volume to about 60%.

Adjust the Tx power if needed (F/W | TxPO | Turn collar knob on top of radio to desired power setting).

I was able to connect to an RMS without any other changes.

Appendix III: Installation Script

If you are in a hurry to get your Winlink station up and running, I wrote a bash script that installs all of the required applications and dependencies. It is named *pat_winlink_install.sh*. You can find it in the zip file that contained this document.

To run the script, copy three files to your Pi (I put scripts in the ~/Downloads folder).

1. pat_winlink_install.sh
2. files.txt
3. apps.txt

Open a terminal window and make your working directory ~/Downloads.

```
$ cd ~/Downloads
```

Make the script executable:

```
$ chmod +x pat_winlink_install.sh
```

Run the script:

```
$ sudo ./pat_winlink_install.sh
```

It will take a while. The script tells you what it is doing as it progresses. If the script runs into any trouble, it will error out and tell you what happened. This should guide you on how to fix it.

NOTE: The script will not run unless files.txt and apps.txt files are in the same directory.

Once the script has completed its tasks, reboot your Pi.

Appendix IV: Pat-Winlink Startup Script

To save you the trouble of having to manually start Direwolf and PAT-Winlink, I wrote a bash script that does this for you. It is named `start_winlink.sh`. You can find it in the zip file that contained this document.

To run the script, copy it to your home folder (e.g. `~/home/n1spw`).

Open a terminal window and make sure you are in your home directory.

```
$ cd ~/
```

Make the script executable:

```
$ chmod +x start_winlink.sh
```

Create a link to the script on your Desktop. From a terminal window do the following:

```
$ cd ~/
```

```
$ nano start_winlink.desktop
```

Enter the below text

```
[Desktop Entry]
```

```
Type=Application
```

```
Name=start_winlink
```

```
Exec=/home/<your_account_name>/start_winlink.sh
```

```
Terminal=true
```

```
<Ctrl>s <Ctrl>x
```

```
$ chmod +x start_winlink.desktop
```

NOTE: There is a sample Desktop file included in the zip file. Copy it to your Desktop folder and change the Exec path.

Here is what my launcher looks like:

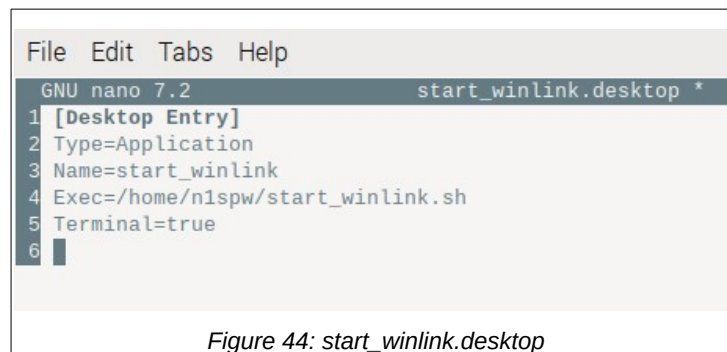


Figure 44: `start_winlink.desktop`

All you need to do to start PAT-Winlink is to click on the `start_winlink` icon on your desktop, then click on *Execute*.