AuroraWatchNet Raspberry Pi magnetometer manual

Steve R. Marple,
Lancaster University.

2017-08-22 23:08:57 +0100
Commit: e3be795d07169eeddc15f4c73bd6543e2c88c6ea

Licence

Contents

List of figures

Abbreviations

API	application programming interface
DHCP	dynamic host configuration protocol
DNS	domain name system
DTR	data terminal ready
FAT	file allocation table
GPIO	general purpose input/output
GPU	graphics processing unit
HMAC	hash-based message authentication code
HTTP	hypertext transfer protocol
HTTPS	hypertext transfer protocol secure
I2C	inter-integrated circuit (bus)
IEEE	Institute of Electrical and Electronics Engineers
IP	internet protocol
LED	light emitting diode
MD5	message digest 5
NFS	network file system
NTP	network time protocol
PoE	power over ethernet
SD	secure digital
SSH	secure shell
TCP	transmission control protocol
UDP	user datagram protocol
URL	uniform resource locator
USB	universal serial bus
UTC	coordinated universal time

Part I

Introduction

Chapter 1

Overview of the hardware

1.1 Introduction

The magnetometer is designed for low-power operation, simple installation and ease of construction. The entire design is open source, allowing anyone with reasonable soldering ability to construct one.

The magnetometer has two parts, the interfacing unit, which is fitted onto the Raspberry Pi's GPIO expansion header, and the sensor head. Both parts should be located outdoors, away from buildings, cars and other sources of human disturbance.

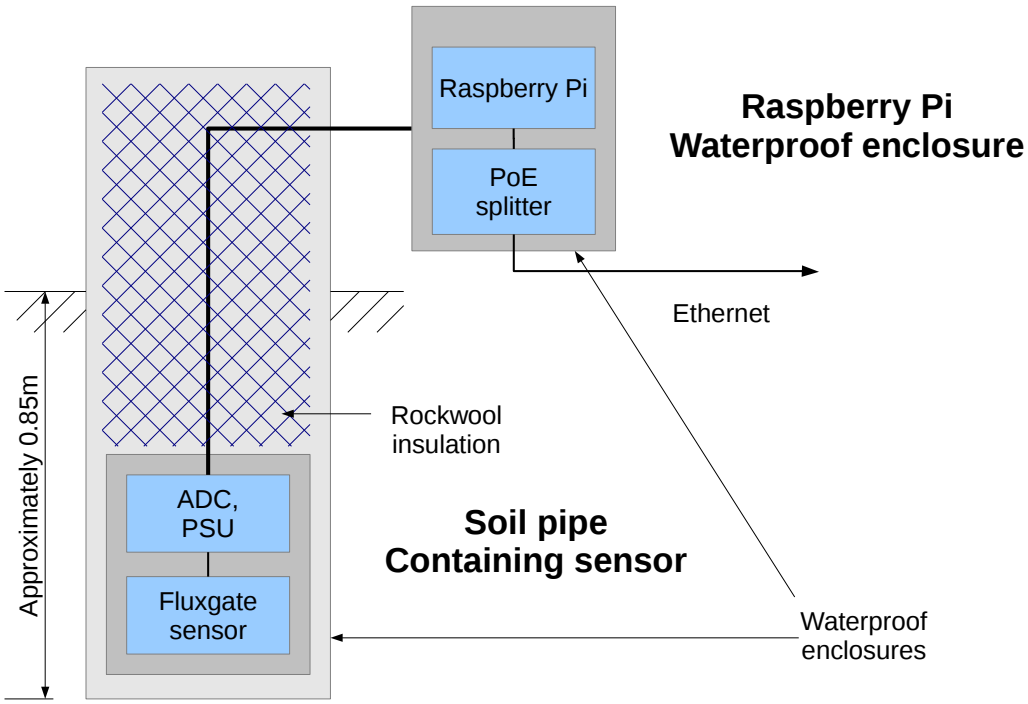

Figure 1.1: System overview.

Part II

Construction

TO DO

Part III

Installation

Chapter 2

Site requirements

2.1 Sensor requirements

The sensor should be located outside away from human distrubances. The site for the sensor should be chosen with regard to the following requirements, with the highest priority given first.

- Away from moving metal objects, for example, trains (more than 50 m), cars (more than 20 m) and garage doors.
- Away from static metal objects, in particular those containing the *ferro-magnetic* materials iron, nickel and cobalt.

The FLC100 fluxgate magnetometer sensor is slightly sensitive to temperature variations. To ensure correct behaviour a stabilised temperature environment is required. This is obtained by burying the sensor.

2.2 Raspberry Pi requirements

The Raspberry Pi requires power and a network communication. The normal mode of operation is to utilise a IEEE 802.3af standard PoE network. Alternatives such as passive PoE are possible but not recommended.

2.2.1 Network requirements

The following network requirements are needed for the system to operate fully:

- DNS resolution. This is normally provided as standard on most networks.
- Outgoing access on port 80 (HTTP) and 443 (HTTPS). Required for software updates and data transfer to AuroraWatch UK and the Met Office Weather Observations Website.

- Outgoing access on port 123 (NTP), or access to a local NTP server.
- Outgoing SSH access (port 22) is required if `rsync` or `rrsync` data transfers are enabled.

Chapter 3

Raspberry Pi setup

3.1 SD card creation

If your SD card already contains Raspbian you can skip to section 3.2.

Download the latest Raspbian image and copy to the SD card following the instructions on the Raspberry Pi web site. **Copying the compressed image to a FAT partition on the SD card will not work**.

These instructions assume Debian 8 ("Jessie") is used. Most users should probably download the desktop version. Advanced users, and those planning to use the Pi without a display connected ("headless") should probably download the minimal version without the desktop software.

3.2 Configuring Raspbian

> In the command window you can press the Tab key to have Linux complete the command or filename.

> If you are not familiar with the `nano` text editor read The Beginner's Guide to Nano, the Linux Command-Line Text Editor

Raspbian is most easily configured by booting the new image. If you are able to discover the IP address (for instance, by checking the DHCP tables of your home router) you can do this over the network using SSH. Otherwise you must use attach a keyboard and monitor to the Raspberry Pi. If you are familiar with Linux it is also possible to edit the files by mounting the SD card on another Linux system.

3.2.1 Raspbian configuration

Log in as `pi` and run

```
sudo raspi-config
```

3.2.1.1 Change user password

If the default password has not been changed then do so now to keep your system secure.

3.2.1.2 Hostname

If you wish change the hostname of your Raspberry Pi select `Advanced Options` and then `Hostname`.

If you have multiple Raspberry Pi computers on your network then you should arrange for them to have unique hostnames. Our preference is to set the hostname to `awn-xxx`, where `xxx` is the abbreviation (typically 3 letters) used for the magnetometer site.

3.2.1.3 Boot options

Configure the Pi for console access (text console, requiring the user to login). The console can be started manually when required with the `startx` command which saves memory when an interactive login is not required.

3.2.1.4 Localisation options

`cron` uses local time and the shift to and from daylight saving time complicates the `cron` tables. Set the Raspberry Pi's timezone to UTC to avoid daylight saving.

Select `Localisation Options` and then `Change Timezone`. For geographic area select `None of the above`, then select `UTC`.

3.2.1.5 Interfacing options

Enable the SSH server so that you can log in remotely.

Enable the Arm I2C interface.

3.2.1.6 Memory split

Select `Advanced Options` and then `Memory Split`. Set the GPU memory to `16` (MB).

3.2.1.7 SSH server

If you plan to log into the Raspberry Pi remotely you should enable the SSH server. If you will only log in via keyboard and monitor connected directly to the Pi then you can choose to disable the SSH server. Select `Advanced Options` and then `SSH`. Choose the appropriate option.

3.2.1.8 Expand Filesystem

The filesystem should be expanded to use all of the (micro)SD card. Advanced users planning to make a backup of the card may wish to perform this step last so that the backup can be smaller.

Select `Advanced Options` and then `Expand Filesystem`. Choose `Finish` and then reboot.

3.2.2 Configure proxy server

Not all networks require a proxy server (or web cache) to be used, your network adminstrator should be able to advise. If it is necessary the setting should be configured in two places.

As user `root`

```
nano /etc/environment
```

At the end of the file add a line similar to

```
http_proxy='http://proxyhost:port/'
https_proxy='http://proxyhost:port/'
```

You must replace `proxyhost` and `port` with the correct settings for your network. If the proxy server requires a username and password the lines should be similar to

```
http_proxy='http://username:password@proxyhost:port/'
https_proxy='http://username:password@proxyhost:port/'
```

Replace `username` and `password` with the values your network administrator has provided.

Repeat for the procedure, as `root`

```
nano /etc/apt/apt.conf.d/10proxy
```

Add a line similar to

```
Acquire::http::Proxy "http://proxyhost:port";
```

Or, if a password is required, similar to

```
Acquire::http::Proxy "http://username:password@proxyhost:port";
```

A separate line for HTTPS is not required in `/etc/apt/apt.conf.d/10proxy`.

Proxy settings will not take effect until you log out and log back in. Type

```
logout
```

and then log back in.

3.2.3 Upgrade installed software

As user `root`

```
apt-get update
apt-get upgrade
apt-get dist-upgrade
```

3.2.4 Remove swap file

To prolong the life of the SD card a swap file is not used. As user `root`

```
apt-get remove dphys-swapfile
```

3.2.5 Remove Wolfram Engine

Wolfram Engine uses over 680 MiB and is not needed. Remove to save valuable space on the SD card. As user `root`

```
apt-get purge wolfram-engine
apt-get autoremove
```

ℹ Wolfram engine is not installed by default in the light version of Raspbian.

3.2.6 Install missing software packages

As user `root`

```
apt-get install screen git git-doc git-man \
    python-pip ipython python-matplotlib \
    python-scipy python-serial python-daemon python-lockfile \
    avahi-daemon dnsutils i2c-tools python-smbus python3-smbus ntp
```

3.2.7 Configure file system mount options

As user `root`

```
nano /etc/fstab
```

Find the line where the root file system is mounted, it will look similar to

```
/dev/mmcblk0p2  /               ext4    defaults,noatime  0       1
```

Change the mount options (`defaults,noatime` in the example above) so that the mount options are now `noatime,nodiratime`. The line should look similar to the one below.

```
/dev/mmcblk0p2  /               ext4    noatime,nodiratime  0       1
```

At the end of the `/etc/fstab` add the following lines:

```
# tmpfs for AuroraWatchNet temporary files. Files will be deleted on
# a reboot, which is desirable for the NTP status files.
tmpfs  /home/pi/tmpfs  tmpfs  rw,size=100k,nr_inodes=1k,noexec,nodev,nosuid,uid=pi,gid=pi,mode=1700  0  0
```

As user `root`

```
mkdir /home/pi/tmpfs
mount /home/pi/tmpfs
```

3.2.8 Automatically create symlinks for FTDI all-in-one

As user `root`

```
nano /etc/udev/rules.d/90-usb_serial.rules
```

Insert the following lines into the file if they are not present:

```
# Have symlinks based on serial number for FTDI devices. Used for
# awnetd_monitor
SUBSYSTEMS=="usb", KERNEL=="ttyUSB[0-9]*", ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001", SYMLINK+="tty_ftdi_%s{serial}"
```

3.2.9 Regenerate SSH host keys

As user `root`

```
cd /etc/ssh
rm ssh_host_*_key{,.pub}
ssh-keygen -A
```

3.3 Installing the AuroraWatchNet server software

3.3.1 Install the Git repository

As user `pi`

```
git clone --recursive https://github.com/stevemarple/AuroraWatchNet.git
git clone --recursive https://github.com/stevemarple/auroraplot.git
git clone --recursive https://github.com/stevemarple/python-MCP342x.git
mkdir ~/bin
. ~/.bashrc
cd ~/bin
ln -s ../AuroraWatchNet/software/server/bin/raspimagd.py
ln -s ../AuroraWatchNet/software/server/bin/log_ip
ln -s ../AuroraWatchNet/software/server/bin/upload_data.py
cd ~
```

3.3.2 Create data directory

As user `root`

```
mkdir /data
chown pi.pi /data
```

3.3.3 Create configuration file

As user `root`

```
cp ~pi/AuroraWatchNet/software/server/ini_files/raspimagd_awnet.ini /etc/awnet.ini
nano /etc/awnet.ini
```

In the editor find the `[DEFAULT]` section, edit the `site` to the correct value. Navigate to the `[upload]` section and enter the correct values for `url`, `username`, `password` and `realm`. Ensure the leading comment character (`#`) is removed.

3.3.4 Configure `cron`

As user `pi`

```
crontab ~/AuroraWatchNet/software/server/crontabs/raspimagd.crontab
```

If you plan to use the FTDI-all-in-one programmer and switch to indicate periods of bad data then uncomment the line with the reference to `awnetd_monitor.py`.

If the Raspberry Pi is using Wi-Fi networking uncomment the line referencing `network_watchdog`; it will cause the Raspberry Pi to be rebooted if it appears that Wi-Fi networking has stopped working for some reason.

The `log_ip` lines in the cron file cause the Raspberry Pi to periodically report that it is operating. This helps the AuroraWatch administrators monitor which stations are active. These reporting commands can be omitted if you prefer (comment out the command by inserting a `#` at the start of the line).

3.3.5 Configure `ifplugd`

Configure `ifplugd` to report when the network interface has been assigned an IP address, which helps the AuroraWatch administrators monitor which stations are active. This step can be omitted if you prefer. As user `root`

```
cd /etc/ifplugd/action.d
ln -s /home/pi/AuroraWatchNet/software/server/bin/log_ip
```

3.3.6 Configure python

Create the python local site directory and create appropriate symbolic links. As user `pi`

```
~/AuroraWatchNet/software/server/bin/setup.py --sudo
~/AuroraWatchNet/software/server/bin/setup.py --sudo
~/AuroraWatchNet/software/server/bin/setup.py --sudo
```

The first time the command runs there will be some errors printed as symbolic links are other configuration details are missing. On the second run confirm that the errors have been corrected.

3.3.7 Configure `ntp`

Check that the current time is correct by typing

```
date --utc
```

This will output the current date and time in UTC. If you aren't certain what the current time in UTC is then you can check at this web page, `https://www.google.co.uk/#q=utc+time`.

Check that the NTP service is running correctly:

```
~pi/bin/check_ntp_status --log-level=info
```

The last line should indicate "NTP synchronized". If it indicates that NTP is not synchronized consult your network manager for the correct NTP settings on your network.

3.4 Backup of SD card and expand filesystem

If you did not expand the (micro)SD card earlier then make a backup (if you wish) and run

```
sudo raspi-config
```

Select `Expand Filesystem`. Choose `Finish` and then reboot.

Chapter 4

Installation procedure

4.0.1 Tools required

- Spade.
- Fork.
- Small bucket or other container to remove soil from the bottom of the hole.
- Compass.

The following items are optional but if they are available they may be useful for digging the hole.

- Soil auger.
- Post hole digger.

4.1 Network infrastructure

TO DO

4.2 Base unit installation

Connect the Raspberry Pi to wired ethernet connection. Connect the keyboard, mouse and monitor (if using) before powering up the Raspberry Pi. Connect the Raspberry Pi to a 5 V power supply with an output current rating of at least 700 mA. If you are not using a monitor connect to the Raspberry Pi using SSH, the default hostname is `raspberry.local`.

4.3 Installing the sensor unit

The sensor unit must be should be buried to a depth of 0.85 m, if it is buried too shallow the unit will be more susceptible to temperature variations. After digging a suitable hole install the enclosure as vertical as possible and backfill the hole. Insert the wooden frame and rockwool into the enclosure. Using a compass align the north arrow on the wooden frame to point towards magnetic north. TO DO: Complete...

Chapter 5

Contributing data

5.1 Contributing data to AuroraWatch UK

5.1.1 Use of data by AuroraWatch UK

Data contributed to AuroraWatch UK will be combined with other magnetometer data for the purpose of generating AuroraWatch UK or other auroral-related alerts. In future the alerts data are likely to be made available via a public API under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 license. Ideally you will also license the magnetometer data under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 license, remembering to define your own attribution requirements.

> If you choose a more permissive license, such as without the attribution, and/or non-commercial clauses, but retain the share-alike clause you must grant AuroraWatch UK permission to use the data to generate alerts under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 license. This is because the share-alike clause restricts others from imposing additional restrictions.

The magnetic field data collected by AuroraWatch UK magnetometers will be made publically available under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 license, with a short embargo period (24 to 48 hours). If your magnetometer data is also licensed as CC BY-NC-SA 4.0 then AuroraWatch UK will share your data in the same way.

5.1.2 Methods to upload data

Two methods to upload data to AuroraWatch UK are supported, using rsync through an SSH tunnel, or by HTTP. Rsync contains an algorithm to efficiently transfer only the differences btween the local and remote files and thus is ideal for transferring the real-time data files. SSH is used to provide a secure connection method. However, SSH access may not be possible on some

networks (e.g., school networks). For cases when rsync cannot be used a HTTP upload method is available which emulates some of the behaviour of rsync; whenever possible only the latest additions to a file are uploaded. The upload method is normally defined in `/etc/awnet.ini` configuration file, in the `[upload]` section.

5.1.2.1 Rsync uploads

As user `root` edit `/etc/awnet.ini`. At the end of the file add the `[upload]` section if it is missing, it should appear as

```
[upload]
method = rsync
```

As user `pi` create the keys for public key authentication:

```
ssh-keygen -t dsa
```

When prompted for the filename to save the key press  to accept the default. When prompted for the passphrase press  for an empty passphrase. Keep the private key (`/home/pi/.ssh/id_dsa`) secret, send the public key (`/home/pi/.ssh/id_dsa.pub`) to AuroraWatch UK.

Create the SSH config file to define hostname and user used for data transfer. As user `pi` edit the file `/home/pi/.ssh/config`, it should look similar to

```
Host awn-data
Hostname uploadhost
User uploaduser
```

You will need to obtain the upload hostname and username from AuroraWatch UK.

Insert an instruction into the `crontab` file to upload the data at regular intervals. As user `pi`:

```
crontab -e
```

Add the following lines:

```
### rsync upload
*/3 * * * * nice /home/pi/bin/upload_data.py > /dev/null 2>&1
```

5.1.2.2 HTTP uploads

As user `root` edit `/etc/awnet.ini`. At the end of the file add the `[upload]` section if it is missing, it should appear as

```
[upload]
method = http
url = upload_URL
realm = upload_realm
password = upload_password
```

You will need to obtain the upload URL, realm and password from AuroraWatch UK.

Insert an instruction the the `crontab` file to upload the data as regular intervals. As user `pi`:

```
crontab -e
```

Add the following lines:

```
### HTTP upload
# Upload text data for today at regular intervals
*/5 * * * * nice /home/pi/bin/upload_data.py -s today --file-types awnettextdata
# Make several attempts to upload all files from yesterday
5 */6 * * * nice /home/pi/bin/upload_data.py -s yesterday > /dev/null 2>&1
```

5.2 Contributing data to the Met Office

Data can be contributed to the Met Office Weather Observations Website (WOW). First create a user account in WOW, see `https://register.metoffice.gov.uk/WaveRegistrationClient/public/register.do?service=weatherobservations`. Then register your site (`https://wow.metoffice.gov.uk/sites/create`) to obtain a site ID and 6 digit authorisation key.

The relevant information must be added to the `/etc/awnet.ini` file. As user `root`:

```
nano /etc/awnet.ini
```

Add the following lines:

```
# Upload to Met Office WOW
[wow]
url = <upload URL>
site_id = <site ID>
site_auth = <authorisation key>
```

Replace `<upload URL>` with the appropriate URL for magnetometer data. (This information has not yet been published). Replace `<site ID>` with the 36 character site indentifier and `<authorisation key>`) with the 6 digit authorisation key.

Data can be uploaded manually by running the following command as user `pi`:

```
/home/pi/AuroraWatchNet/software/server/bin/upload_wow.py -c /etc/awnet.ini -s yesterday
```

ℹ The WOW API does not provide a means to check if the data has been uploaded previously, and the WOW site does not check for or remove duplicate data. It is therefore necessary for the user to check if data has been already uploaded. For this reason it is recommended that data is uploaded only data recording to a given file has completed, i.e., avoid uploading data from the current day.

It is possible to automate the data uploads with a cron job. As user `pi`:

```
crontab -e
```

At the end of the crontab add the following lines:

```
# Upload data to Met Office WOW
5 * * * * /home/pi/AuroraWatchNet/software/server/bin/upload_wow.py -c /etc/awnet.ini -s yesterday > /dev/null 2>&1
```

Part IV

Operation

Chapter 6

Raspberry Pi operation

6.1 Introduction

For generic operation of the Raspberry Pi (setting the hostname, assigning a fixed IP address etc.) please see the Raspbian documentation, `http://www.raspbian.org/RaspbianDocumentation`.

6.2 Shutting down the Raspberry Pi

The Raspberry Pi must be shutdown cleanly before power is removed:

```
sudo shutdown -h now
```

Before removing the power wait until only the red power LED is lit; wait a further two seconds to ensure futher access to the SD card is not needed. If the power is removed whilst data is being written to the SD card it will corrupt the file system.

To reboot the Raspberry Pi use

```
sudo shutdown -r now
```

6.3 Starting and stopping the data recording daemon

Data is recorded on the Raspberry Pi using a *daemon* process, which is started and stopped by the Debian init scripts. The scripts must be started and stopped as user `root`, the actual data

recording process runs as user `pi`.

To start data recording

```
sudo /etc/init.d/awnetd start
```

To stop data recording

```
sudo /etc/init.d/awnetd stop
```

It is also possible to check the status of the data recording process

```
sudo /etc/init.d/awnetd status
```

The `restart` option forcibly stops recording (if running) and then starts it again:

```
sudo /etc/init.d/awnetd restart
```

6.4 Monitoring the data recording process

The data recording process directs its standard output and error streams to a virtual terminal using `screen`. It is possible to attach to this virtual terminal to monitor the output.

As user `pi`

```
screen -r awnet
```

To exit from `screen` type CTRL - a , d .

Pressing CTRL - c will terminate the recording process.

Chapter 7

Software and firmware updates

7.1 Raspbian updates

See Raspberry Pi web pages, `https://www.raspberrypi.org/documentation/raspbian/updating.md`.

7.2 AuroraWatchNet software updates

The software can be updated easily simply by *pulling* a new version from the Github repository. As user `pi`

```
cd ~/AuroraWatchNet
git pull
```

Appendix A

Configuration file options

A.1 Introduction

Many of the AuroraWatchNet programs read a common configuration file, typically located at `/etc/awnet.ini`. The configuration file is broken into sections, each of which starts with a section header in square brackets (`[like_this]`). Other lines contain key names and values, written as `key = value`. Leading and trailing whitespace around both the key and value is ignored. Section headers and key names do not contain whitespace, words may be separated with an underscore. The configuration file can also contain comments, which are entered with a hash (`#`) or semi-colon (`;`) as the first character.

The configuration file is parsed using Python's `SafeConfigParser` module. This allows values defined in the same section, or in the `[DEFAULT]` section to be inserted into other key value definitions. This feature is commonly used to insert the site abbreviation into filenames.

A.2 `[DEFAULT]`

The `[DEFAULT]` section is special as it defines values which can used elsewhere in the configuration file.

A.2.1 project

Define the project. This is used elsewhere within the configuration file, e.g., data filenames.

Default: none.

A.2.2 site

Define the site code, typically a three letter abbreviation. This is used elsewhere within the configuration file, e.g., data filenames.

Default: none.

Example:

```
site = lan1
```

A.3 [awnettextdata]

Options associated with the standard text-format output data file.

A.3.1 filename

Define the filename used for text-format data files. This string is expanded as a `strftime` format string and accepts the normal `strftime` format specifiers; for a list of the acceptable format specifiers see `https://docs.python.org/2/library/time.html#time.strftime`. However, since Python expands the string first any percent characters used as part of a `strftime` format specifier must be repeated.

Default: none.

Example:

```
filename = /data/aurorawatchnet/%(site)s/%%Y/%%m/%(site)s_%%Y%%m%%d.txt
```

`%(site)` is replaced with the site abbreviation which was defined previously in the `[DEFAULT]` section. Notice how the `strftime` format specifiers require two % characters.

For June 20th 2014 with the site abbreviation `cwx` this would expand to
`filename = /data/aurorawatchnet/cwx/2014/06/cwx_20140620.txt`

A.4 [awpacket]

Options associated with the standard binary output format. This format is incovenient to read but preserves the received data messages from the magnetometer, and the responses sent back from the recording daemon. It is possible to play back these files to the recording daemon and regenerate other data formats.

A.4.1 filename

See section A.3.1 for a description.

Default: none.

Example:

Typically set to
```
filename = /data/aurorawatchnet/%(site)s/%%Y/%%m/%(site)s_%%Y%%m%%d.awp
```

A.4.2 key

By default the binary data packets are written out with their original signing key. If you plan to make the binary data format available then for security reasons you should probably set a different signing key. You will then be able to share this key without compromising the communication channel with the magnetometer and others will be able to use the error-checking capabilities provide by HMAC-MD5. The key is a 32 character hexadecial string, without any `0x` prefix.

Default: none.

Example:

Typically set to a simple code,
```
key = 00000000000000000000000000000000
```

A.5 [logfile]

Options associated with the recorded log files.

A.5.1 filename

See section A.3.1 for a description.

Default: none.

Example:

Typically set to
```
filename = /data/aurorawatchnet/%(site)s/%%Y/%%m/%(site)s_%%Y%%m%%d.log
```

A.6 [daemon]

Options relating to the data-recording daemon.

A.6.1 connection

Defines the connection between the daemon and the magnetometer. Radio communication emulates a serial port connection. If the magnetometer is PoE model the communication should be set to `ethernet`.

Default: `serial`

A.6.2 close_after_write

Have the daemon close the data and log files after each write. It is recommended that this option only be used on an NFS file system, when used on a file sytem store on flash memory (such as the Raspberry Pi) this may cause excessive writes and early failure of the flash memory. This option may be required for real-time data collection daemons where the data files are also received by `rsync` or `HTTP` uploads.

A.6.3 acknowledge

If set to `false` then acknowledgements are not sent in response to magnetometer data messages.

Default: `true`

A.6.4 read_only

If set to `true` then for serial connections the device is opened read-only and no set up commands are sent.

Read-only mode implies that acknowledgements are not sent, regardless of the setting of the `acknowledge` option.

Default: `false`

A.7 [controlsocket]

It is possible for the `send_cmd.py` program to send commands to the magnetometer via the data recording daemon. Communication is via a UDP socket or a unix domain socket.

A.7.1 filename

Use a unix domain socket for communication with the data recording daemon, with the given filename. If set to `none` then a control socket will not be created. If the `filename` option is present it takes priority over any `port` option.

Default: none.

A.7.2 port

Use a UDP socket for communication with the data recording daemon, with the given port number. If set to `none` then a control socket will not be created. If the `filename` option is present it takes priority over the `port` option.

Default: `6587`

Example:

```
port = 6587
```

A.8 [magnetometer]

Settings associated with the magneometer.

A.8.1 siteid

The numeric site identifier for the magnetometer. The recording daemon will ignore data packets where the site ID does not match the value set in the configuration file. The site ID should be set as an integer number in the range 0 to 255 inclusive.

Default: none.

A.9 [upload]

Configuration details for the `upload_data.py` program.

A.9.1 method

Define the upload method in use. Valid options are: `rsync`, `rrsync`, `http`, and `https`. `rsync` uses the `rsync` program to efficiently transfer only the portions of data files which have changed

since the last upoad; `rrsync` is similar but the server restricts which directories may be written to. Both are tunnelled through SSH and thus require the network to permit outward TCP connections to port 22.

The `http` and `https` upload methods make use of the standard HTTP(s) protocol, and therefore require outgoing TCP connections to port 80 and 443 respectively.

A.9.2 `rsync_host`

Name of the `rsync` host. No facility to set the user is provided, use the SSH `config` file to set the appropriate details.

Default: none.

Example SSH `config` file:

```
Host awn-data
Hostname host.domain.com
User monty
```

A.9.3 `url`

The URL to which HTTP//HTTPS uploads are sent.

A.9.4 `password`

The upload password for HTTP and HTTPS methods. Digest authentication is used.

A.9.5 `username`

The upload username for HTTP and HTTPS methods. If not specified it defaults to the site abbreivation prefixed with `awn-`.

A.9.6 `realm`

The *realm* used for digest authentication when uploading data with the HTTP or HTTPS method.

A.10 [realtime_transfer]

The data recording daemon is capable of forwarding the incoming binary packets to one or more remote hosts. When this is enabled data is transferred in real-time. The remote host(s) can use the same data recording daemon but should be configured to be read-only so that acknowledgements are not sent.

When data packets are forwarded in this way the daemon does not check that they have been delivered successfully. It is recommended that this method is used in conjunction with the `upload_data.py` program to ensure any data packets which were not delivered are transferred by some other means.

Real-time transfer uses three keys, `remote_host`, `remote_port` and `remote_key`. To send data to multiple hosts a suffix is applied to each of this keys to group the settings together, e.g., `remote_host2`, `remote_port2` and `remote_key2`. The suffix must not contain whitespace characters.

A.10.1 remote_host

The hostname or IP address to which data packets are sent.

A.10.2 remote_port

The UDP port to which data packets are sent.

A.10.3 remote_key

The HMAC-MD5 key used to sign the data packets. This should be different to the key used to communicate with the magnetometer.

A.11 [dataqualitymonitor]

The data recording daemon can monitor for the existence of a file indicating that data quality may be compromised (e.g., due to local site activities such as cutting grass). If the file is present an extension is appended to the data files to clearly separate the poor quality data. Any real time data transfers in operation are suspended. When the file is removed the normal filenames are used again and real time data transfer resume as normal.

A.11.1 port

Defines the serial port used by the `awnetd_monitor.py` daemon. If using a USB device it is recommended to modify the `udev` rules so that a fixed device name is generated based on the serial number.

> Example `udev` rule to create a symbolic link for an FTDI USB serial adaptor, save to `/etc/udev/rules.d/90-usb_serial.rules` or similar:
>
> ```
> # Have symlinks based on serial number for FTDI devices. Used for
> # awnetd_monitor
> SUBSYSTEMS=="usb", KERNEL=="ttyUSB[0-9]*", \
> ATTRS{idVendor}=="0403", ATTRS{idProduct}=="6001", SYMLINK+="tty_ftdi_%s{serial}"
> ```

Default: none.

A.11.2 filename

The name of the file which when present indicates reduced data quality. This file can be created by the `awnetd_monitor` daemon or an external process.

Default: `/var/aurorawatchnet/data_quality_warning`

A.11.3 extension

The extension which is appended to files to indicate reduced data quality.

Default: `.bad`

A.11.4 pidfile

A file containing the process ID of the `awnetd_monitor` daemon.

Default: none.

A.11.5 logfile

The file where the log output of the `awnetd_monitor` daemon is written.

Default: none.

A.11.6 led_active_low

A flag indicating if the DTR signal line should be pulled low to turn on the LED.

Default: true.

A.12 [ntp_status]

Configuration details for data recording daemon (`awnetd.py`) and also used by the `check_ntp_status` program. As the Raspberry Pi does not have an onboard real-time clock it must obtain the time from the network using NTP. At boot time, and other occasions when the NTP servers are not accessible, the system clock on the Raspberry Pi may not be correct. The data recording daemon can be made aware that NTP is running and correctly synchronized by the presence of a semaphore file. The file's age is also checked to ensure that the presence of a stale semaphore file is not misinterpreted. When the data recording daemon is configured to check NTP status the current time is sent to the microcontroller only if NTP is synchronized.

A.12.1 filename

The filename of the semaphore file. Required for `check_ntp_status`. Not required by `awnetd.py` and if missing NTP status is not checked.

A.12.2 max_age

The maximum allowable age in seconds for the semaphore file to be considered valid. The presence of a file older than this age is ignored. Required for `check_ntp_status`. Not required by `awnetd.py` and if missing NTP status is not checked.