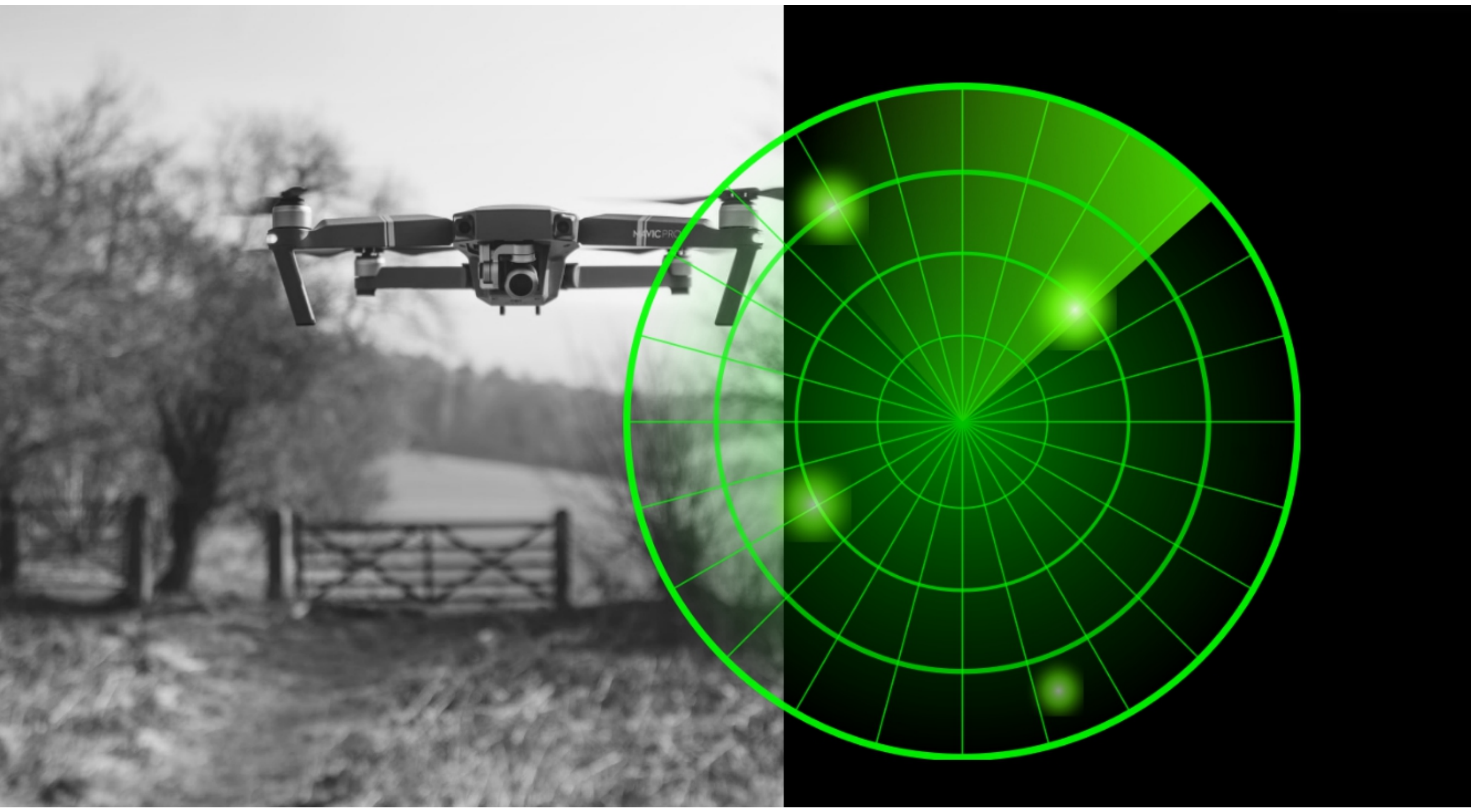
DroneScout - Sensor Manual

November 2021 - version 1.1



The latest version of this manual is located here:
https://download.bluemark.io/dronescout_sensor_manual.pdf

Intended audience: system integrators

Disclaimer: we are not responsible or liable for errors or incomplete information in this document.

Version history

version	date	description
1.0	April 2021	● Initial release
1.1	November 2021	● Updated document with the features of the latest firmware release

Contents

1 Introduction

1.1 Audience

This document is intended for system integrators that want to use the *DroneScout* sensor in their own product.This document is not intended for normal users!

1.2 Sensor

The sensor runs on Armbian 21.02 Linux distribution (ARM64). It can be accessed via SSH on the local network.

Login

The login credentials are configured per system integrator/client. Please contact us for the password.

User name: root
Password: ask
Service: SSH
Port: 22
IP address: DHCP client in local network

Read-only file system

The sensor uses a so-called overlayroot file system. The file system is mounted read-only and there is a read-write file system in memory on top of it. This means that the system can be used normally, but after a reboot all changes to the file system are lost. Using a read-only file system prevents for instance corrupt file systems after an unexpected power loss.

To make changes permanent:

- Enter in the SSH console: `overlayroot-chroot`
  - After this command you can make changes to the filesystem. Enter `exit` to exit this mode.
- Or mount the read-only partition as read-write instead. I.e enter `mount -o remount,rw /media/root-ro`
  - Make the changes in the folder /media/root-ro
  - Remount as read-only: `mount -o remount,rw /media/root-ro`

Reboot afterwards.

2 Configuration

2.1 mqtt.conf

In /root/mqtt.conf, the mqtt broker settings are stored. The default contents is:

```
#MQTT settings

host="portal.dronescout.co"
port="10094"

username="" # default empty, not used
password="" # default empty, not used

QoSlevel="1" # QoS level, 0 or 1 or 2, use 0 for highest capacity!

data_dump_interval="10" #interval in seconds for dumping data to the mqtt
broker.

#SSL config
ssl="1" #set 1 for SSL connection, 0 for connections without encryption
CAfile="/root/.ssh/rootCA.crt" #root certificate #root certificate for
SSL connection
CertificateFile="/root/.ssh/certificate.crt" #certificate for SSL
connection
PrivateKeyFile="/root/.ssh/private.key" #private key for SSL connection
extra_options="--insecure" # use option "--insecure" if using
self-generated certificates
```

To change the MQTT broker, adjust the settings in this file accordingly.

2.2 bm.conf

The /root/bm.conf file defines general settings. Currently, it only contains settings for reverse SSH connections.

```
maintenance="1" # enable remote SSH connections
maintenance_server="portal.dronescout.co"
maintenance_port="10090"
maintenance_user="bluemark"
```

2.3 sensorid.cfg

The file /root/sensorid.cfg configures the sensor ID. It is a numerical number without the DS prefix.

2.4 login.cfg

The file /root/login.cfg configures the login credentials for the web-interface. If the file does not exists, the default credentials are admin with password bluemark.

The file contents is: <username>:<password>

2.5 threshold_override.conf

The file /root/threshold_override.conf allows to set a minimum detection threshold per 1-MHz bin. This is useful to suppress phantom alarms for instance caused by nearby WiFi networks. The format per line is: <1-MHz bin number>, <threshold in dBm>.

For 2.4 GHz band, the 1-MHz bin number is 0 to 79 and for 5.8 GHz it is 80 to 209 and 5.2 GHz it is 210 to 319. The default threshold for 2.4 GHz is -94.0 dBm and for 5.8 GHz + 5.2 GHz -117.1.

Examples:

- the line *0, -94.0* defines that for 1-MHz bin number 0 (frequency 2402.5 MHz, the minimum signal threshold is -94 dBm)
- the line *70, -92.3* defines that for 1-MHz bin number 70 (frequency 2472.5 MHz, the minimum signal threshold is -92.3 dBm)
- the line *80, -114.0* defines that for 1-MHz bin number 80 (frequency 5725.5 MHz, the minimum signal threshold is -114.0 dBm)
- the line *201, -115.1* defines that for 1-MHz bin number 80 (frequency 5846.5 MHz, the minimum signal threshold is -115.1 dBm)
- the line *210, -115.1* defines that for 1-MHz bin number 210 (frequency 5145.5 MHz, the minimum signal threshold is -115.1 dBm)

For maximum sensitivity, use a value of -100 for the 2.4 GHz and for 5.8 GHz + 5.2 Ghz, -120. Use this command to change all lines in threshold_override.conf

```
overlayroot-chroot
sed  -i 's/-94.0/-100.0/' threshold_override.conf
sed  -i 's/-117.1/-120.0/' threshold_override.conf
exit
reboot -f
```

Note that in firmware version 20211115-1215 and higher, this file can be edited more convenient via the web-interface. Also the actual thresholds of the sensing are saved in the file /root/threshold.conf. It has the same format as the threshold_override.conf file and is updated every minute. After 1 hour it reaches full sensitivity and those values can be used to configure the threshold_override.conf settings.

2.6 reverse SSH

When sensors are placed behind a router, they can't be accessed remotely. Reverse SSH is a method to be able to login remotely without changing router or firewall settings. See https://www.howtogeek.com/428413/what-is-reverse-ssh-tunneling-and-how-to-use-it/ for background information. If reverse SSH is enabled (by setting maintenance to “1” in bm.conf), the sensor will try to connect to the configured SSH server (using IP address, port and user) and opens a port that can be used to remotely login. The used port is calculated as follows: sensor ID modulo 50000 + 10000. For example, for sensor ID 204 the port will be 10204.

In addition, passwordless login by using SSH keys is assumed. This means that the sensor SSH key (/root/.ssh/id_rsa) needs to be accepted by the remote SSH server. More information can be found in the mentioned background article.

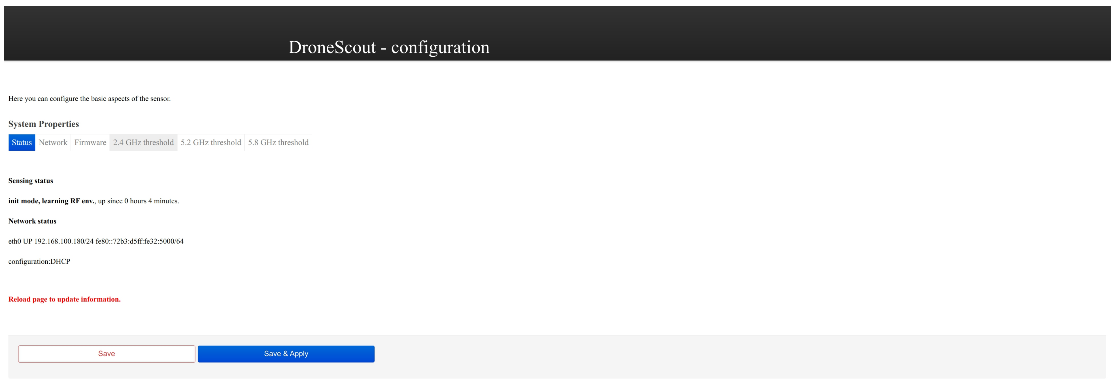
Login from the SSH server

The sensor can be accessed from the SSH server by entering the following command: `ssh root@localhost -p10204`

The number 10204 is the port selected by the sensor. In this example, it is the port for sensor ds000204. For sensor ds000100, the port would be 10100.

2.7 Web-interface

In firmware version 20211115-1215 and higher, the sensor has a web-interface to configure the main settings of the sensor: update firmware, configure the network-settings and configure the threshold_override.conf file. Enter the IP-address of the sensor in your browser to access it. Please see the user manual for more information.



Here you can configure the basic aspects of the sensor.

System Properties

Sensing status

init mode, learning RF env., up since 0 hours 4 minutes.

Network status

eth0 UP 192.168.100.180/24 fe80::72b3:d5ff:fe32:5000/64

configuration:DHCP

Reload page to update information.

Figure 1 - Screenshot of the web-interface.

3 MQTT MESSAGES

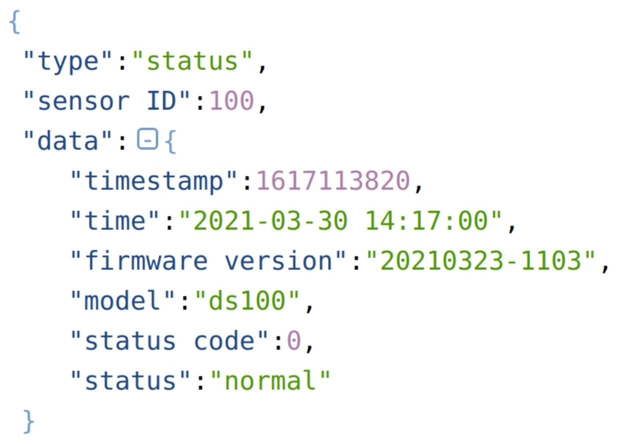
The sensor generates multiple realtime alarms (JSON format):

- *Sensor status* - every minute the status of the sensor
- *Realtime alarm* - every 10 seconds, the system will generate an alarm for currently detected drones
- *Alarm summary* - Five minutes after a drone has been detected, an alarm summary message is generated.

Alarms are published to the MQTT broker configured in /root/mqtt.conf

Sensor status

The portal will publish every minute a sensor status message. A example message is shown below.



```
{
 "type":"status",
 "sensor ID":100,
 "data":{
    "timestamp":1617113820,
    "time":"2021-03-30 14:17:00",
    "firmware version":"20210323-1103",
    "model":"ds100",
    "status code":0,
    "status":"normal"
 }
```

Figure 2 - Example sensor status message.

This message contains several fields:

- *type* - "status" for this type of alarm
- *sensor ID* - the ID of the sensor
- *data* - the data section contains: the time in unix timestamp and UTC time, the firmware version, the model and the status both in code and text. The status can be: 0 - normal, 1 - temperature too high, 2 - invalid license, 3 - init mode, learning RF env. If the system is too hot, the load on the system is temporarily reduced to prevent overheating. If the sensor is rebooted, the first hours are used to learn the RF environment. For status code 1 and 3 the performance of the system is reduced; a bit less sensitive. In case of status code 2, the detection is disabled.

Realtime alarm

When a drone is detected, the sensor will generate every 10 seconds an alarm. An example alarm is shown below.

```
{
 "type":"alarm",
 "sensor ID":100,
 "data":{
    "timestamp":1617113810,
    "time":"2021-03-30 14:16:50",
    "type":"unknown",
    "frequency MHz":5759.0,
    "bandwidth MHz":2.0,
    "rssi dBm":-114
 }
```

Figure 3 - Example realtime alarm message.

This message contains several fields:

- *type* - “alarm” for this type of alarm
- *sensor ID* - the ID of the sensor
- *data* - the data section contains: the time in both unix timestamp and UTC time. Furthermore, the frequency and bandwidth are included, both in MHz values. Based on the bandwidth the type of the drone is determined. It can be *WiFi*, *proprietary* or *unknown*. Also, the current received signal strength (rssi) is included in dBm value.

Alarm summary

An alarm summary message is generated 5 minutes after the last detection of the drone. An example message is shown below.

```
{
 "type":"alarm summary",
 "sensor ID":100,
 "data":{
    "timestamp start":1617108995,
    "timestamp stop":1617109064,
    "time start":"2021-03-30 12:56:35",
    "time stop":"2021-03-30 12:57:44",
    "type":"proprietary",
    "frequency MHz":2469.0,
    "bandwidth MHz":4.0,
    "max rssi dBm":-63
 }
```

Figure 3 - Example alarm summary message.

This message contains several sections:

- *type* - “alarm summary” for this type of alarm
- *sensor ID* - the ID of the sensor
- *data* - the data section contains: the start and stop detection time in both unix timestamp and UTC time. Furthermore, the frequency and bandwidth are included, both in MHz values. Based on the bandwidth the type of the drone is determined. It can be *WiFi*, *proprietary* or *unknown*. Also, the maximum received signal strength (rssi) is included in dBm value.

4 Log RSSI information

In firmware version 20211115-1215 and higher, it is possible to log the internal threshold and detected RSSI power per 1 MHz frequency block. Enabling this option will consume roughly 60 MB of RAM per hour. Furthermore, the sensor has up to 5 hours RAM storage. After that the sensor will stop functioning.

Enabling this option can be done as follows:

```
touch /root/output_RSSI_enabled
killall dronescout
```

After a reboot this file is gone and the sensor acts normally. For a permanent logging of RSSI power data, do a `overlayroot-chroot` first to store this file permanently.

After a few minutes a file will be created in the folder /root/output/ that ends on ".rssi.csv". After finishing the measurements, kill the dronescout process: `killall dronescout` and copy this file to your computer. This can be done via SCP command on the computer, not sensor!:

```
scp root@<IP address sensor>:/root/output/*.rssi.csv .
```

This command will copy all RSSI files to the current folder.

The CSV contains the following columns:

1. Epoch time seconds
2. Epoch time milliseconds part
3. 1 MHz frequency bin
4. RSSI average value
5. RSSI standard deviation value
6. Current RSSI value
7. RSSI threshold for this 1 MHz bin
8. RSSI standard deviation threshold for this 1 MHz bin

Example row:
1629980824,949,2402.5,-73.61,2.64,-73.67,-65.80,7.92

The measured RSSI values can be plotted with the following Matlab or Octave script. Octave is a free open source clone of Matlab and can be downloaded here:
https://www.gnu.org/software/octave/download

It will plot every 1 MHz RSSI line versus time for the 3 different frequency bands.

```
clear all
close all

#here we assume the csv is called out.csv and
#located in /tmp. Change accordingly.
wlan=importdata('/tmp/out.csv');

figure
for channel = 0:79
   hold on
   I = find(wlan(:,3) == 2402.5+ 1*channel);
   plot(wlan(I(50:end),4),'LineWidth',2)
end
grid minor on
grid minor on
xlabel("time")
ylabel("RSSI")
title("2.4 GHz")

figure
for channel = 1:100
   hold on
   I = find(wlan(:,3) == 5145.5+ 1*channel);
   plot(wlan(I(50:end),4),'LineWidth',2)
end
grid minor on
grid minor on
xlabel("time")
ylabel("RSSI")
title("5.2 GHz")

figure
for channel = 1:129
   hold on
   I = find(wlan(:,3) == 5725.5+ 1*channel);
   plot(wlan(I(50:end),4),'LineWidth',2)
end
grid minor on
xlabel("time")
ylabel("RSSI")
title("5.8 GHz")
```

5 FIRMWARE UPDATE

The firmware can be updated using the web-interface.
.

6 MORE INFORMATION

If you need more information, please contact us at info@bluemark.io or by phone: +31 53 711 2104.

All contact information can be found at the *DroneScout* contact page:
https://dronescout.co/contact/