Aerobits

Subsystems for the
UAS intergration into
the airspace

Ground Station with Linux

Data sheet & User manual

Introduction

GS with Linux station is an **ADS-B** and **FLARM** Omni-directional receiver station with **BLE/Wi-Fi** RemoteID receiver and also multi-constellation **GNSS** sensor on board to provide best accuracy. **LTE** connectivity allows usage in all LTE/4G rich environments without the need for any additional cabling to send data. It has been designed to allow quick and easy assembly, enclosed in IP67 case for high weather condition resistance. Device comes with all necessary cables and antennas for straightforward installation.

Single-board computer with **Linux** operating system gives incredible expansion possibilities, much higher computing power and memory, way easier updates, quicker debugging and development times.

Supply voltage is provided using **PoE** Power over Ethernet, using regular ethernet cable between PoE power supply and GS Linux. Depending on cable quality range can be up to 100m, giving it a tremendous advantage over USB. At the same time it can still transfer data just like a regular ethernet cable, providing an alternative data connection.

It is a perfect solution for permanent installation in open areas for constant **airspace monitoring** and conducting VLOS/BVLOS operation where safety is critical.

NOTE: The device to operate on FLARM frequency requires FLARM UAS license. The license can be obtained with the device from Aerobits upon purchase. FLARM library expire after year and must be updated with latest firmware.

Applications

- Airports and critical infrastructure
- Nationwide traffic management systems (manned and unmanned)
- Perfect solution for local airfields
- U-Space and UTM systems
- Ground Network air traffic surveillance systems
- Network based Remote Identification (central monitoring)

For more information please contact: support@aerobits.pl.

Contents

1 Technical parameters

1.1 Basic technical information

Parameter	Description	Typ.	Unit
First Band	ADS-B	1090	MHz
Second Band	FLARM	868	MHz
Third Band	BLE	2400	MHz
Fourth Band	Wi-Fi	2400	MHz
Sensitivity (ADS-B)		-94	dBm
Sensitivity (FLARM)		-109	dBm
Sensitivity (BLE)		-103	dBm
Sensitivity (Wi-Fi)		-103	dBm
Ethernet (RJ45)	Standard Ethernet 10/100		
LTE Cat. 1	Data transport layer (global bands)		
Enclosure	Ingress protection	IP67	

Table 1: General technical parameters.

1.2 Electrical specification

1.2.1 Power supply

Parameter	Value
Power connector	Standard ethernet connector (power supply and optionally network) and Bulgin Buccaneer 400 Series connector (power supply PX0412/03P)
Power consumption	3.5 W
Power supply	100 - 240 VAC with PoE supply unit or 24 VDC with AC/DC converter

Table 2: Power supply of GS-LINUX

1.3 Mechanical specification

1.3.1 Mechanical parameters

Parameter	Value
Dimensions	170 x 170 x 100 mm
Weight	0.5 kg (Module without cables and antennas) 2.5 kg (With cables, arm and antennas)

Table 3: Mechanical parameters of GS-LINUX

2 Protocols

2.1 JSON Protocol

Each message is encoded as separate JSON object, without any excess whitespace, consisting of fields described in table 4.

Name	Description	Value type
`{`		
`"src": "ID-0000001",`	Ground Station with Linux serial number.	String
`"ts": 69061337,`	Timestamp in milliseconds, relative to last UTC midnight. Value `69061337` encodes 19:11:01.337. Omitted if unknown.	Unsigned integer
`"ver": 1,`	JSON protocol version. See details below.	Unsigned integer
`"gnss": {...}`	One or more of the data fields, described in subchapters below.	Object or array
`}`		

Table 4: Description of main JSON fields.

NOTE: The order of JSON object fields in any part of message may vary between firmware revisions and messages.

Some JSON objects have fields, of which values may sometimes be unknown. In this case, they are skipped in JSON output. In following chapters, each of those fields are explicitly marked as ommitable.

NOTE: In case of JSON objects consisting of only ommitable fields, if none of them are set, the whole object may be omitted.

The `"ver"` field indicates JSON protocol version. Future ICD versions may introduce additional fields without changing the version number. If a breaking change occurs in Ground Station with Linux JSON specification, the version number is guaranteed to be incremented.

NOTE: The version number of JSON protocol described in this document is 1.

2.1.1 GNSS section

The `"gnss"` section contains basic GNSS information. This message is sent once per second. The example JSON message with `"gnss"` section fields described, is shown in table 5.

JSON field	Description	Value type
`{`		
`"src": "ID-0000001",`	see table 4.	
`"ts": 69061337,`		
`"ver": 1,`		
`"gnss": {`		
`"fix": 1,`	Set to 1 if onboard GNSS currently has fix, otherwise 0.	Unsigned integer
`"lat": 53.42854,`	Last known latitude. Omitted if there was no GNSS fix since device boot.	Floating point
`"lon": 14.55281,`	Last known longitude. Omitted if there was no GNSS fix since device boot.	Floating point
`"altWgs84": 499.6,`	Last known WGS-84 Altitude, in meters. Omitted if there was no GNSS fix since device boot.	Floating point
`"altMsl": 508.6,`	Last known MSL Altitude, in meters. Omitted if there was no GNSS fix since device boot.	Floating point
`"track": 127.3,`	Track angle, 0°..360°, relative to true north. Omitted if unknown.	Floating point
`"hVelo": 10.5,`	Horizontal velocity, in knots. Omitted if unknown.	Floating point
`"vVelo": 25.00,`	Vertical velocity, in m/s. Positive value is upwards. Omitted if unknown.	Floating point
`"gndSpeed": [`		
`5.2, 2.1`	Ground speed in east-west and north-south axes respectively, in knots. Positive value is East and North. Derived from `track` / `hVelo` values. Omitted if unknown.	Floating point
`],`		
`"acc": {`		
`"lat": 5.2,`	Accuracy of latitude, in meters. Omitted if unknown.	Floating point
`"lon": 2.1,`	Accuracy of longitude, in meters. Omitted if unknown.	Floating point
`"alt": 3.6`	Accuracy of altitude, in meters. Omitted if unknown.	Floating point
`},`		
`"nacp": 12`	Navigational Accuracy Category for Position value, as defined in ED-282. Omitted if unknown.	Unsigned integer
`"nacv": 2`	Navigational Accuracy Category for Velocity value, as defined in ED-282. Omitted if unknown.	Unsigned integer
`"nic": 12`	Navigation Integrity Category as defined in ED-282. Omitted if unknown.	Unsigned integer
`}`		
`}`		

Table 5: Descriptions of JSON GNSS section fields.

2.1.2 Sensor section

The "sensor" section contains values acquired from miscellaneous sensors present in Ground Station with Linux hardware and consists of fields shown in table 6. This message is sent once per second. All fields are optional - they are sent only if appropriate sensor is enabled.

JSON field	Description	Value type
{		
"src": "ID-0000001",	see table 4.	
"ts": 69061337,		
"ver": 1,		
"StationParams": {		
"Ina219A": {		
"BusVoltage": 4.6,	Current voltage sensor value in V.	Floating point
"BusCurrent": 650.122,	Current current sensor value in mA.	Floating point
"Power": 3000.573,	Current power sensor value in mW.	Floating point
},		
"SHT40": {		
"Temperature": 36.9,	Current temperature sensor value in C°.	Floating point
"Humidity": 19,	Current humidity sensor value in %.	Unsigned integer
},		
"MS5837": {		
"Pressure": 1000.5,	Current pressure sensor value, in hPa.	Floating point
"Temperature": 36.9,	Current temperature sensor value in C°.	Floating point
"MSLRelativeAltitude": 38.15,	Current MSL value based on pressure, in meters.	Floating point
} } } }		

Table 6: Descriptions of JSON sensor section fields.

2.1.3 LTE section

The "lte" section contains values acquired from LTE dongle in Ground Station with Linux hardware and consists of fields shown in table 7. This message is sent once per second. Not all data are important, for more information visit LTE dongle manufacturer page.

JSON field	Description	Value type
{		
"src": "ID-0000001",	see table 4.	
"ts": 69061337,		
"ver": 1,		
"StationLteDongle": {		
"signal": {		
"pci": "40",	Visit LTE dongle manufacturer page	String
"sc": "",	Visit LTE dongle manufacturer page	String
"cell_id": "4327740",	Visit LTE dongle manufacturer page	String
"rsrq": "-11.0dB",	Reference signal received quality	String
"rsrp": "-96dBm",	Reference signal receive power	String
"rssi": "-67dBm",	Received signal strength indicator	String
"sinr": "5dB",	Signal to interference plus noise ratio	String
"rscp": null,	Visit LTE dongle manufacturer page	String
"ecio": null,	Visit LTE dongle manufacturer page	String
"mode": "7",	Visit LTE dongle manufacturer page	String
"ulbandwidth": "15MHz",	LTE uplink bandwidth	String
"dlbandwidth": "15MHz",	LTE dowlink bandwidth	String
"txpower": {		

"PPusch": 12dBm,	Dongle output power	String
"PPucch": 999dBm,	Visit LTE dongle manufacturer page	String
"PSrs": 999dBm,	Visit LTE dongle manufacturer page	String
"PPrach": 999dBm,	Visit LTE dongle manufacturer page	String
},		
"tdd": null,	Visit LTE dongle manufacturer page	String
"ul_mcs": null,	Visit LTE dongle manufacturer page	String
"dl_mcs": null,	Visit LTE dongle manufacturer page	String
"earfcn": "DL:525 UL:18525",	Visit LTE dongle manufacturer page	String
"rrc_status": "1",	Visit LTE dongle manufacturer page	String
"rac": null,	Visit LTE dongle manufacturer page	String
"lac": null,	Visit LTE dongle manufacturer page	String
"tac": "417",	Visit LTE dongle manufacturer page	String
"band": "1",	Visit LTE dongle manufacturer page	String
"nei_cellid": null,	Visit LTE dongle manufacturer page	String

"plmn": "26006",	Visit LTE dongle manufacturer page	String
"ims": "0",	Visit LTE dongle manufacturer page	String
"wdlfreq": null,	Visit LTE dongle manufacturer page	String
"lteulfreq": "19725",	LTE uplink frequency in MHz	String
"ltedlfreq": "21625",	LTE downlink frequency in MHz	String
"transmode": "TM[4]",	Visit LTE dongle manufacturer page	String
"enodeb_id": "0016905",	Visit LTE dongle manufacturer page	String
"cqi0": null,	Visit LTE dongle manufacturer page	String
"cqi1": null,	Visit LTE dongle manufacturer page	String
"ulfrequency": "1972500kHz",	LTE uplink frequency	String
"dlfrequency": "2162500kHz",	LTE downlink frequency	String
"arfcn": null,	Visit LTE dongle manufacturer page	String
"bsic": null,	Visit LTE dongle manufacturer page	String
"rxlev": null	Visit LTE dongle manufacturer page	String
},		
"info": {		
"DeviceName": "E3372-325",	LTE dongle name	String
"SerialNumber": "W6S7S22C23501476",	LTE dongle serial number	String

"Imei": "213710061379321",	LTE dongle IMEI number	String
"Imsi": "261160037682518",	LTE dongle IMSI number	String
"Iccid": "594802105112728250 85",	LTE dongle ICCID number	String
"Msisdn": null,	Visit LTE dongle manufacturer page	String
"HardwareVersion": "CL5E3372M",	LTE dongle hardware version	String
"SoftwareVersion": "3.0.2.61(H057SP5C983)",	LTE dongle software version	String
"WebUIVersion": "WEBUI 3.0.2.61(W13SP3C7201)",	LTE dongle web UI version	String
"MacAddress1": "11:E2:80:02:1A:B0",	LTE dongle MAC address	String
"MacAddress2": null,	Visit LTE dongle manufacturer page	String
"WanIPAddress": "10.127.198.168",	LTE dongle WAN IP address	String
"wan_dns_address": "89.102.135.20,85.178.121.37",	LTE dongle WAN DNS address	String
"WanIPv6Address": null,	Visit LTE dongle manufacturer page	String
"wan_ipv6_dns_address": null,	Visit LTE dongle manufacturer page	String
"ProductFamily": "LTE",	Visit LTE dongle manufacturer page	String
"Classify": "hilink",	Visit LTE dongle manufacturer page	String
"supportmode": "LTE|WCDMA|GSM",	LTE dongle supported modes	String
"workmode": "LTE",	LTE dongle current mode	String
"submask": "255.255.255.255",	LTE dongle mask address	String

"Mccmnc": "26006",	Visit LTE dongle manufacturer page	String
"iniversion": "E3372-325-CUST 3.0.2.1(C965)",	Visit LTE dongle manufacturer page	String
"uptime": "1758",	Visit LTE dongle manufacturer page	String
"ImeiSvn": "02",	Visit LTE dongle manufacturer page	String
"spreadname_en": null,	Visit LTE dongle manufacturer page	String
"spreadname_zh": null	Visit LTE dongle manufacturer page	String
} } } }		

Table 7: Descriptions of JSON LTE module section fields.

2.1.4 Processed Remote ID reports

The `"remoteID"` section contains aircraft information determined by Ground Station with Linux internal Remote ID processing engine. The messages are encoded as JSON array with at least one entry. Each entry is an object consisting of fields denoted in table 8, if field is unknown will be omitted (empty). Reports for each remoteID aircraft are updated once every second.

JSON field	Description	Value type
`{`		
`"src": "ID-0000001",`	see table 8.	
`"remoteid": [`		
`{`		
`"framePrefix": "B4",`	frame prefix, see description 9	String
`"aircraftID": "18099300000132",`	Aircraft ID represented by string value	String
`"idType": 1,`	ID type see table 12	Unsigned integer
`"uasType": 2,`	Callsign of aircraft, see table 13 .	Unsigned integer
`"lat": 53.42854,`	Latitude in degrees, accuracy 0.6 degree	Floating point
`"lon": 14.55281,`	Longitude in degrees, accuracy 0.6 degree	Floating point
`"height": 1,`	Height based on start up altitude, in meters .	Floating point
`"baroAlt": 17,`	Barometric altitude, in meters.	Floating point
`"geoAlt": 17,`	Geometric altitude, in meters.	Floating point
`"track": 70,`	Track angle, 0°... 360°.	Signed integer
`"hVelo": 10.5,`	Horizontal velocity, in m/s, accuracy 0.1 m/s.	Floating point
`"vVelo": 50,`	Vertical velocity, in m/s, positive value is upwards, accuracy 0.1 m/s.	Floating point
`"statusFlag": 0,`	Operation status.	Unsigned integer
`"operatorId": "AAABBBBBBBBBBBBC-DDD",`	The operator number from local FAA department.	String
`"operatorIdType": 2,`	Specific type of Operator ID.	Unsigned integer
`"operatorLat": 53.42854,`	The operator latitude in degrees, accuracy 0.6 degree.	Unsigned integer
`"operatorLon": 14.55281,`	The operator longitude in degrees, accuracy 0.6 degree.	Unsigned integer
`"operatorLocType": 0,`	The operator location type.	Unsigned integer
`"times": 350,`	Timestamp of the sent frame expressed in seconds since current hour, accuracy 0.1 s-1.5 s	Floating point
`"rssi": -50,`	Signal strength, in dBm	Signed integer

"selfIdType": 1,	Self id type 10	Unsigned integer
"selfId": "Test",	Self ID status	String
"frameType": 15,	Frame type 11	Unsigned integer
"mac": "df:a5:c3:84:78:66",	MAC address	String
}		
]		
}		

Table 8: Descriptions of JSON Remote ID section fields.

Whereby the following prefixes mean:

Frame prefix value	Description
B4	Bluetooth 4.0 (Legacy) frame.
B5	Bluetooth 5.0 frame.
WN	Wi-Fi NaN frame.
WB	WB - Wi-Fi beacon frame.

Table 9: RemoteID frame prefix in JSON protocol.

NOTE: Referring to the ASD-STAN prEN 4709-002 standard, our product displays all the required information (ASD-STAN prEN 4709-002 Table 1 - Data Dictionary), optional data is only available upon special request.

Below is a list off self Id types returned in Self Id value field.

Self Id Type value	Description
0	Text Description.
1	Emergency Description.
2	Extended Status Description.
3–200	Reserved.
201–255	Available for private use.

Table 10: RemoteID UAS Self Id type in CSV protocol.

Below is a list off frame types returned in Frame Type value field.

Id Type value	Description
0	Basic ID.
1	Location.
3	Self ID.
4	System.
5	Operator ID.
15	Packed all in one.

Table 11: RemoteID UAS ID Frame type in CSV protocol.

Below is a list of ID types returned in ID Type value field.

Id Type value	Description
0	None.
1	Serial Number.
2	CAA Assigned Registration ID.
3	UTM Assigned UUID.

Table 12: RemoteID UAS ID Type category values in CSV protocol.

Below is a list of emitter category values returned in ECAT value field.

ECAT value	Description
0	None.
1	Aeroplane.
2	Helicopter or Multirotor.
3	Gyroplane.
4	Hybrid Lift.
5	Ornithopter.
6	Glider.
7	Kite.
8	Free Balloon.
9	Captive Balloon.
10	Airship.
11	Free Fall.
12	Rocket.
13	Tethered Powered Aircraft.
14	Ground Obstacle.
15	Other.

Table 13: RemoteID ID Type category values in CSV protocol.

2.1.5 Processed ADS-B reports

The `"adsb"` section contains aircraft information determined by Ground Station with Linux internal ADS-B processing engine. The messages are encoded as JSON array with at least one entry. Each entry is an object consisting of fields denoted in table 14. Reports for each ADS-B aircraft are updated once every second.

JSON field	Description	Value type
`{`		
`"src": "ID-0000001",`	see table 4.	
`"ts": 69061337,`		
`"ver": 1,`		
`"adsb": [`		
`{`		
`"icao": "DABABE",`	ICAO address, 24-bit value encoded in uppercase hexadecimal, with leading zeros.	String
`"sigStr": -95,`	Signal strength.	Signed integer
`"sigQ": 2,`	Signal quality.	Unsigned integer
`"fps": 5,`	Number of raw Mode-S frames received from aircraft during last second.	Unsigned integer
`"lat": 53.42854,`	Latitude. Omitted if position is unknown.	Floating point
`"lon": 14.55281,`	Longitude. Omitted if position is unknown.	Floating point
`"baroAlt": 1725,`	Barometric altitude, in feet. Omitted if unknown.	Signed integer
`"geoAlt": 1712,`	Geometric altitude, in feet. Omitted if unknown.	Signed integer
`"track": 72.18,`	Track angle, 0°..360°. Omitted if unknown.	Floating point
`"hVelo": 10.5,`	Horizontal velocity, in knots. Omitted if unknown.	Floating point
`"vVelo": 50,`	Vertical velocity, in ft/min, positive value is upwards. Omitted if unknown.	Signed integer
`"ident": "TEST8",`	Callsign, up to 8 chars. Omitted if unknown.	String
`"squawk": "7232",`	Squawk, 8 octal digits. Omitted if unknown.	String
`"ecat": 13,`	Emitter category code, see table 15. Omitted if unknown.	Unsigned integer
`"nacp": 3,`	NAC_P value, as described in ED-102A. Omitted if value is 0 (unknown).	Unsigned integer
`"nacv": 1,`	NAC_V value, as described in ED-102A. Omitted if value is 0 (unknown).	Unsigned integer
`"nicBaro": 1,`	NIC_{BARO} value, as described in ED-102A. Omitted if value is 0.	Unsigned integer
`"nic": 2,`	NIC value, as described in ED-102A. Omitted if value is 0 (unknown).	Unsigned integer
`"surf": 1`	Set to 1 if plane is on earth surface. Omitted if plane is in air or unknown.	Unsigned integer
`}`		
`]`		
`}`		

Table 14: Descriptions of JSON ADS-B section fields.

The emitter category values returned in `"ecat"` field is shown in table 15.

"ecat" value	Description
0	Unknown.
1	Light (below 15500 lbs.).
2	Small (15500 - 75000 lbs.).
3	Large (75000 - 300000 lbs.).
4	High-Vortex Large (aircraft such as B-757).
5	Heavy (above 300000 lbs.).
6	High performance (above 5g acceleration and above 400 knots).
7	Rotorcraft.
8	Reserved.
9	Glider, Sailplane.
10	Lighter-Than-Air.
11	Parachutist, Skydiver.
12	Ultralight, hang-glider, paraglider.
13	Reserved.
14	Unmanned Aerial Vehicle.
15	Space, Trans-atmospheric Vehicle.
16	Reserved.
17	Surface Vehicle - Emergency Vehicle.
18	Surface Vehicle - Service Vehicle.
19	Point Obstacle (includes Tethered Balloons).
20	Cluster obstacle.
21	Line obstacle.

Table 15: ADS-B emitter category values in JSON protocol.

2.1.6 Processed FLARM reports

The `"flarm"` section contains aircraft information determined by Ground Station with Linux internal FLARM processing engine. The messages are encoded as JSON array with at least one entry. Each entry is an object consisting of fields denoted in table 16. Reports for each FLARM aircraft are updated once every second.

JSON field	Description	Value type
`{`		
`"src": "ID-0000001",`	see table 4.	
`"ts": 69061337,`		
`"ver": 1,`		
`"flarm": [`		
`{`		
`"idType": 1,`	Aircraft id type. 0: randomized, 1: ICAO, 2: FLARM.	Unsigned integer
`"id": "DABABE",`	Aircraft id, 32-bit value encoded in uppercase hexadecimal, with leading zeros.	string
`"type": 13,`	Aircraft type, see table 17.	Unsigned integer
`"danger": 1,`	Danger level (1-3). Omitted if 0 (no danger)	Unsigned integer
`"lat": 53.42854,`	Latitude.	Floating point
`"lon": 14.55281,`	Longitude.	Floating point
`"alt": 525,`	Barometric altitude, in meters.	Signed integer
`"track": 72`	Track angle, in degrees.	Unsigned integer
`"hVelo": 50,`	Horizontal velocity, in m/s.	Unsigned integer
`"vVelo": 200,`	Vertical velocity, in m/s.	Unsigned integer
`"movMode": 5,`	Movement mode. 1: stationary, 4: circling right, 5: flying, 7: circling left.	Unsigned integer
`"stealth": 1,`	Set to 1 if target has Stealth flag set, otherwise omitted.	Unsigned integer
`"notrack": 1`	Set to 1 if target has Notrack flag set, otherwise omitted.	Unsigned integer
`}`		
`]` `}`		

Table 16: Descriptions of JSON FLARM section fields.

The list of possible FLARM “Aircraft type” values returned in `"type"` field is shown in table 17.

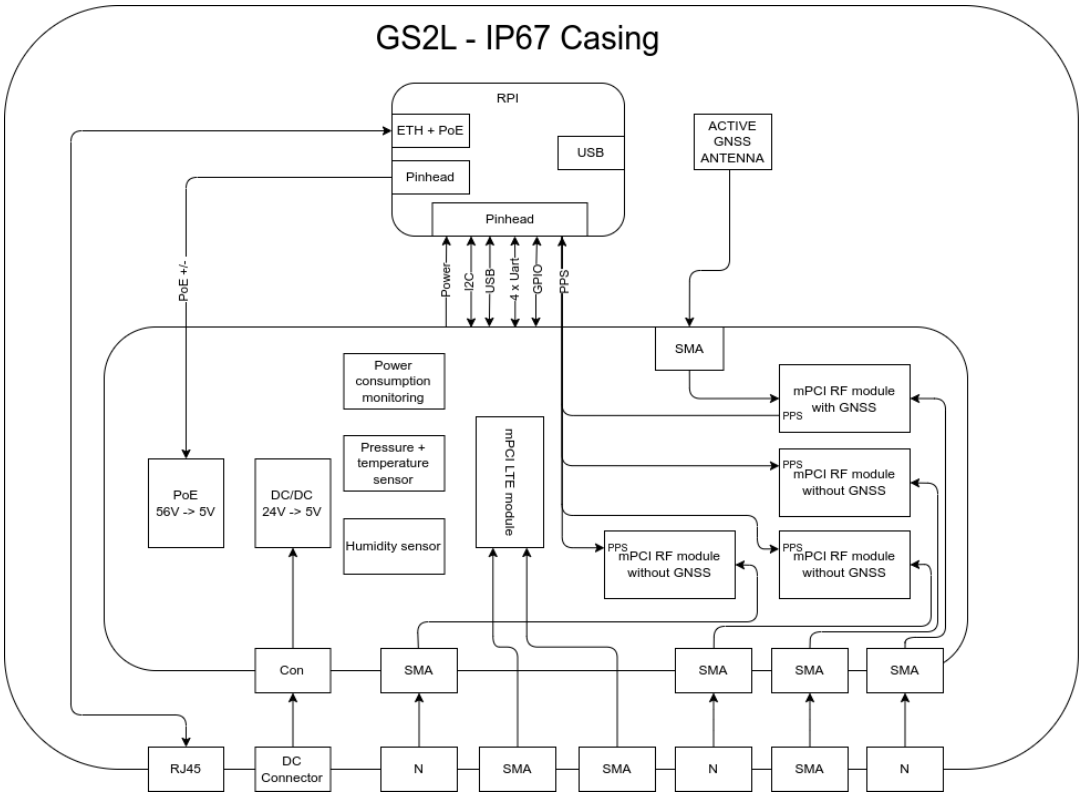
"type" value	Description
0	Reserved.
1	Glider, Motor glider.
2	Tow plane, tug plane.
3	Helicopter, gyrocopter, rotocraft.
4	Skydiver, parachute.
5	Drop plane for skydivers.
6	Hang glider (hard).
7	Hang glider (soft).
8	Aircraft with reciprocating engine.
9	Aircraft with jet / turboprop engine.
10	Reserved.
11	Balloon (hot, gas, weather, static).
12	Airship, blimp, zeppelin.
13	Unmanned Aerial Vehicle (UAV).
14	Reserved.
15	Static obstacle.

Table 17: FLARM Aircraft type values in JSON protocol.

3 GS-LINUX customization

3.1 Diagram

Diagram contains all possible configuration of GS-LINUX, customer can configure ground station on their own.



3.2 Variants

GS-LINUX has multiple configurations, the customer can set up a ground station with several RF modules, up to 4 maximum. Only one module is required to have a GNSS module on board, the rest of the RF modules will receive position information based on the first one. See recommended configuration below.

1. GNSS + RID
2. GNSS + ADS-B + FLARM
3. GNSS + ADS-B + UAT
4. GNSS + RID + ADS-B + FLARM
5. GNSS + RID + ADS-B + UAT

3.3 Additional kits

GS-LINUX has several additional kits such as power delivery, antennas, cables etc.. For example customer can order GS-LINUX with GNSS + RID and DC power supply, so in this case additional need kit with antennas (sector or omnidirectional), power cables, DC converter. Remember GS-LINUX is sold without kits included, kits are sold separate. See list of available kits below.

1. DC Power delivery

2. PoE Power delivery
3. LTE module
4. Sector Antenna with RF cable
5. Omnidirectional Antenna with RF cable
6. AC/DC Power converter with power supply cable
7. Mobile tripod
8. Powerbank (with cables)
9. Mobile box
10. Bracket for 2 omnidirectional antennas

4 Quick start

4.1 Scope of delivery

1. Ground Station with Linux
2. ADS-B/FLARM antenna sector or omnidirectional (optional)
3. BLE/Wi-Fi antenna sector or omnidirectional (optional)
4. Power Supply Cables (optional)
5. Small assembly parts
6. Antenna's installation arm (only with dual omnidirectional antenna setup)
7. Distance bracket
8. Power Supply PoE or AC/DC converter (optional)

Figure 1: GS-LINUX equipment kit

4.2 Installation process

4.2.1 Mounting with sector and omnidirectional antennas

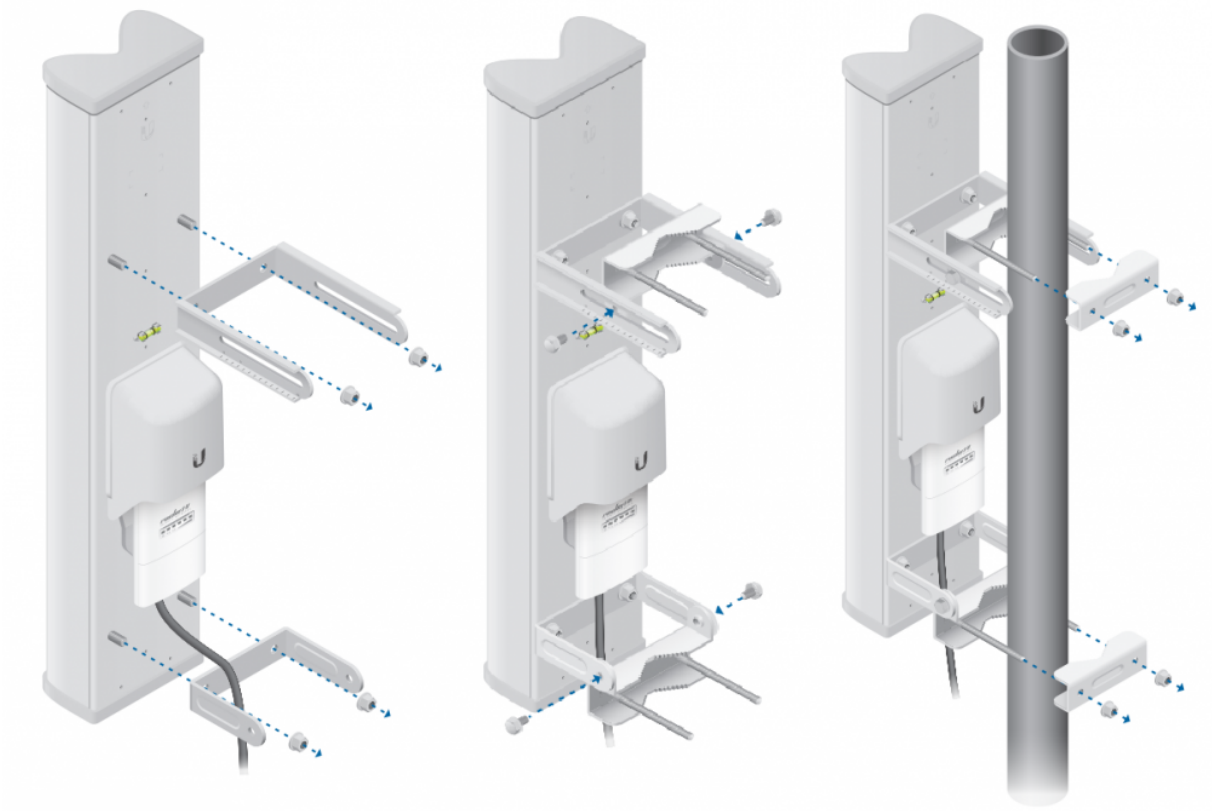
1. Take the GS out of the box and place facing down - as shown on the picture.

2. Mount black distance bracket with the protective earth conductor on the case.

   NOTE: It is important that the cable is connected to the appropriate hole, which is marked on the case with following electrical marking.

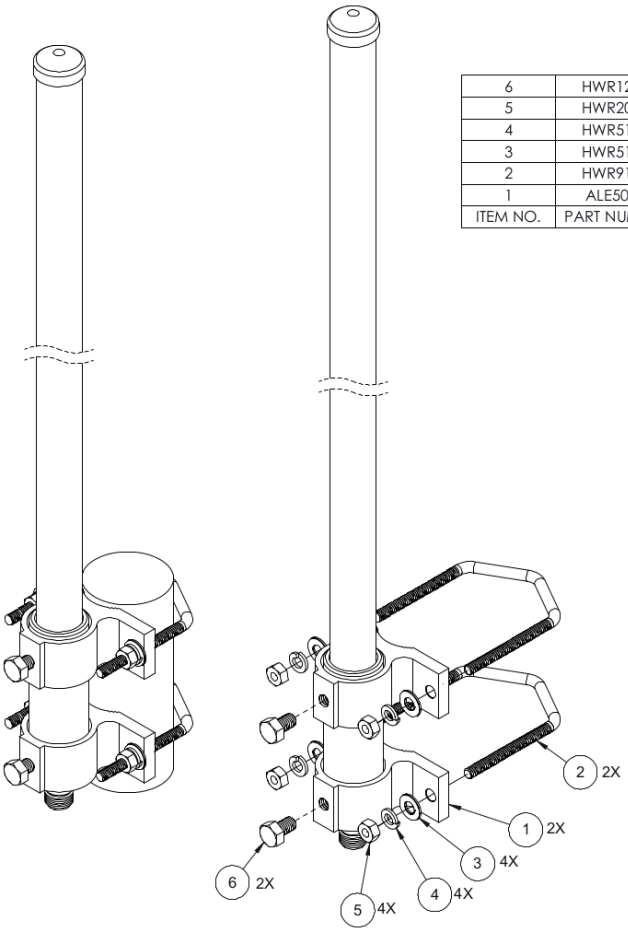
3. After installing black distance bracket, box should looks like this.

4. At last mount antenna, sector:



NOTE: It is important that the cable is connected to the appropriate polarization, for sector antennas Aerobits devices always use vertical polarization (V in antenna outlet description).

5. or omnidirectional:



ITEM NO.	PART NUMBER	DESCRIPTION	QTY
6	HWR1200	BOL,HEX,5/16-18x1/2,SS,NF	2
5	HWR2000	1/4" - 20 NUT	4
4	HWR5100	1/4" - 20 LOCK WASHER	4
3	HWR5150	Washer, Flat for 1/4" Hardware	4
2	HWR9150	CLAMP,VBOLT,1/4-20,ST,NF	2
1	ALE5000	BKT,EXT,AL,FOR FM2	2

6. After the mechanical part of installation, connect the antennas to the device and the device to the ethernet cable as shown 2.

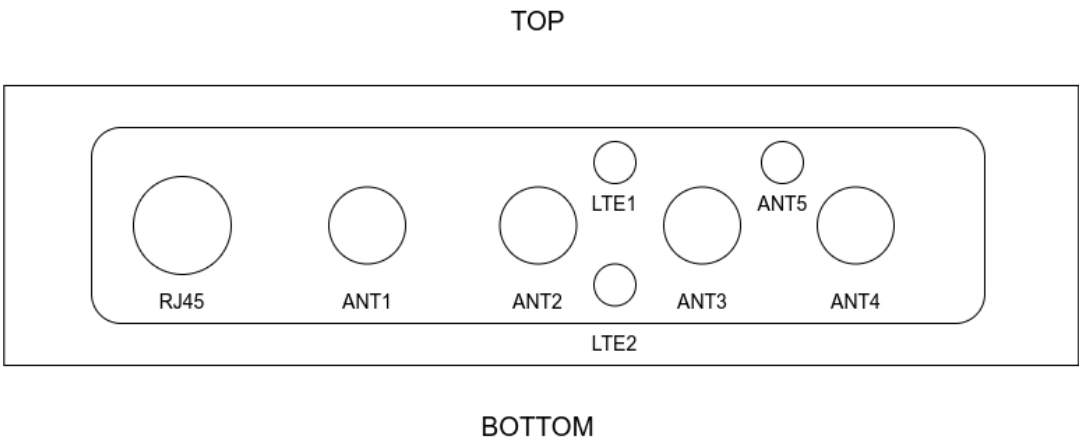
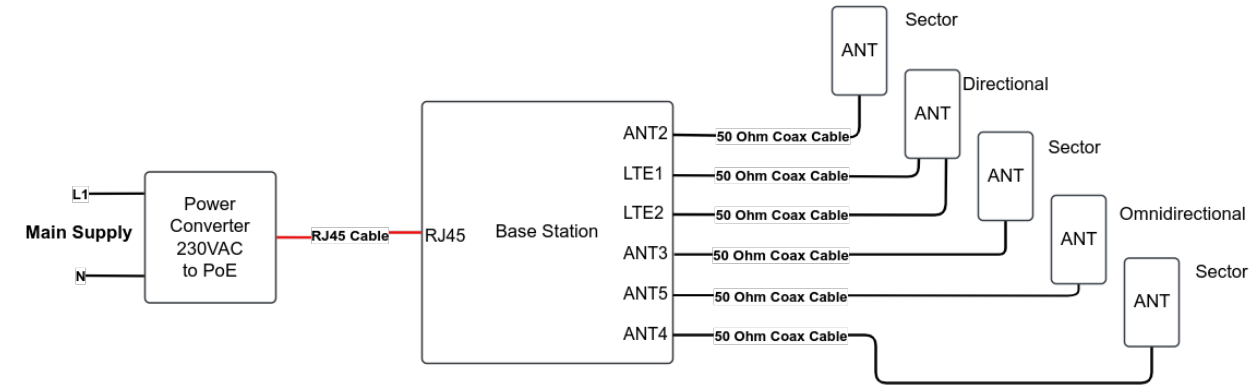


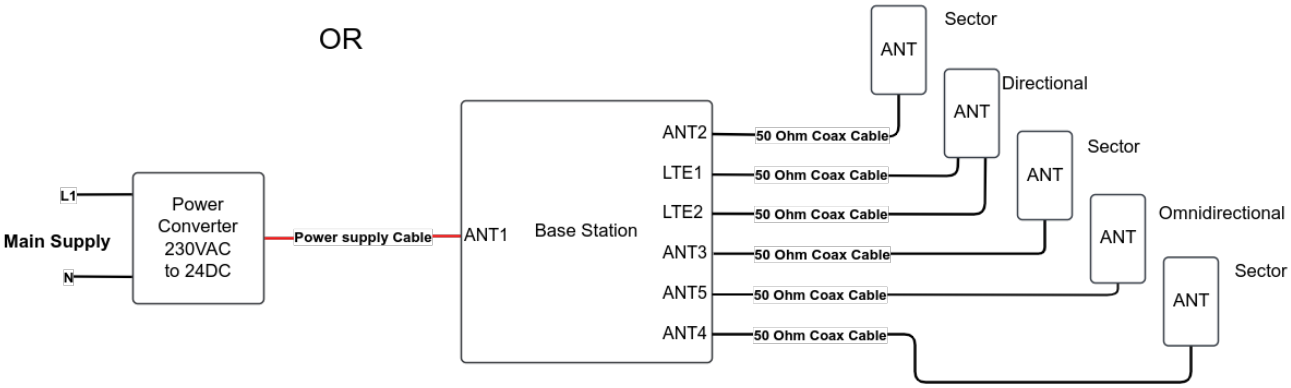
Figure 2: GS-LINUX outlets

1. RJ45 - Ethernet/PoE
2. ANT1 - Power delivery
3. ANT2 - ADS-B/FLARM
4. LTE1 - LTE antenna outlet
5. LTE2 - LTE antenna outlet
6. ANT3 - ADS-B/FLARM
7. ANT5 - RemoteID (BT/Wi-Fi)
8. ANT4 - ADS-B/FLARM

4.2.2 Electrical connection



OR



4.2.3 Power supply connection PoE

First, connect ethernet cable between PoE supply unit OUT socket and GS Linux.

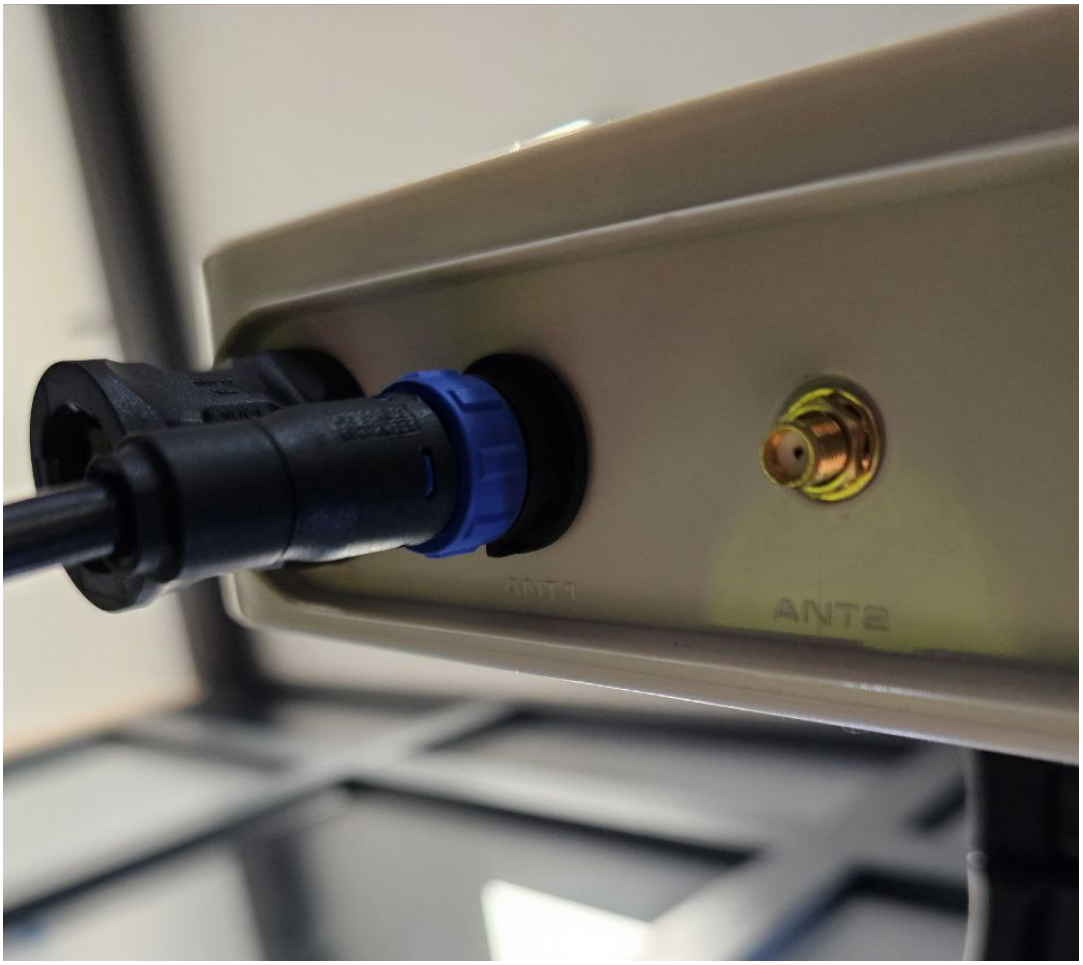
NOTE: This cable will have PoE supply on it, so do not connect to it other devices that cannot handle it.

NOTE: Power up PoE supply only when everything is connected, do not switch connections when supply is on.

Optionally, simply connect other socket labeled IN to your regular router/switch, and GS Linux will be connected to it just as with any regular ethernet connection.

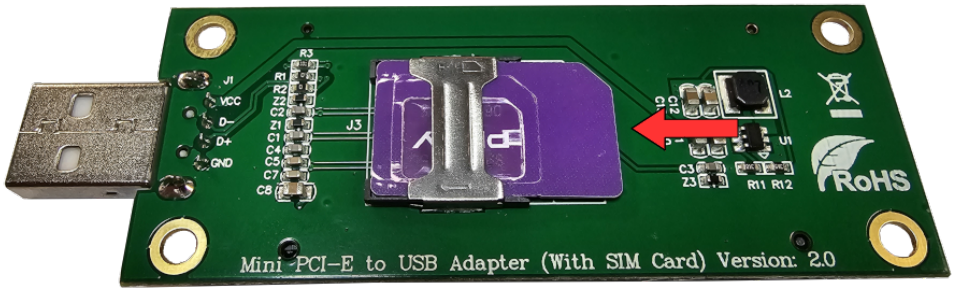
4.2.4 Power supply connection with Bulgin cable

Connect the power cable to the ANT1 socket and supply power with an AC/DC converter.



4.2.5 Inserting a SIM/chip card

Depending on configuration there is a LTE USB stick, that requires a SIM card. Insert card from back just like below.



4.3 Software configuration

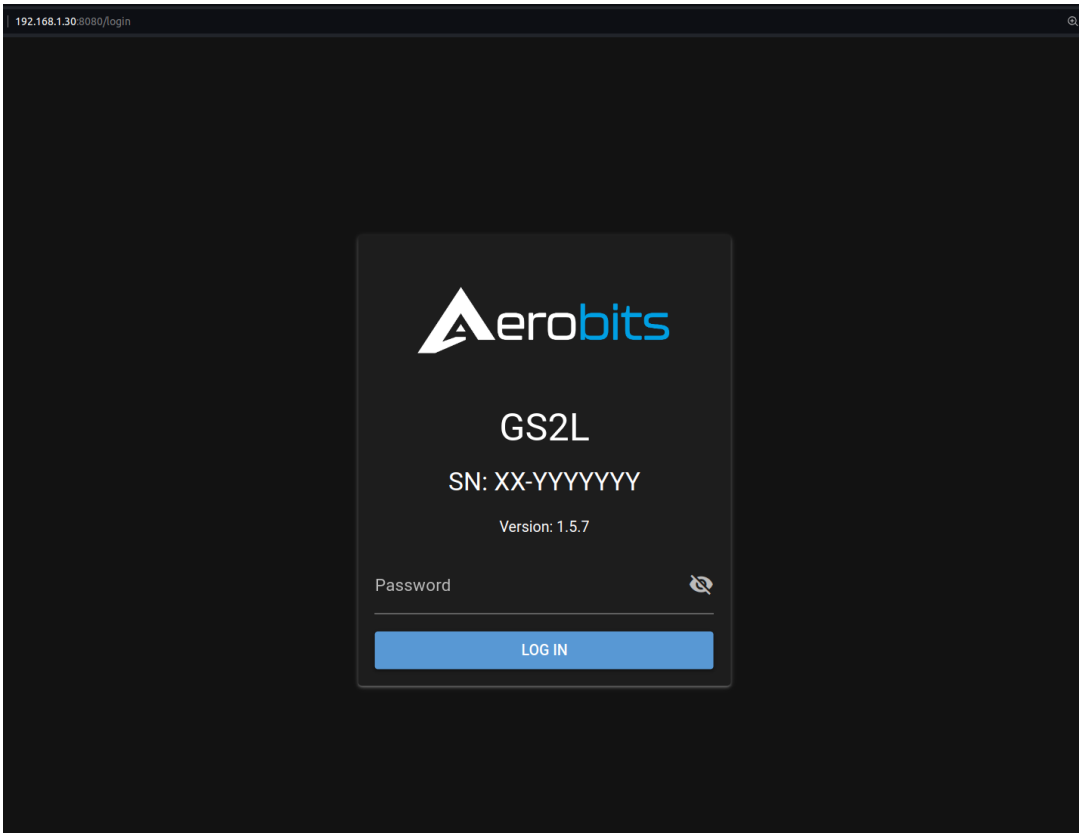
4.3.1 Connection using user interface

Connect station in local network, find its IP address. Start connection using browser and this parameters:

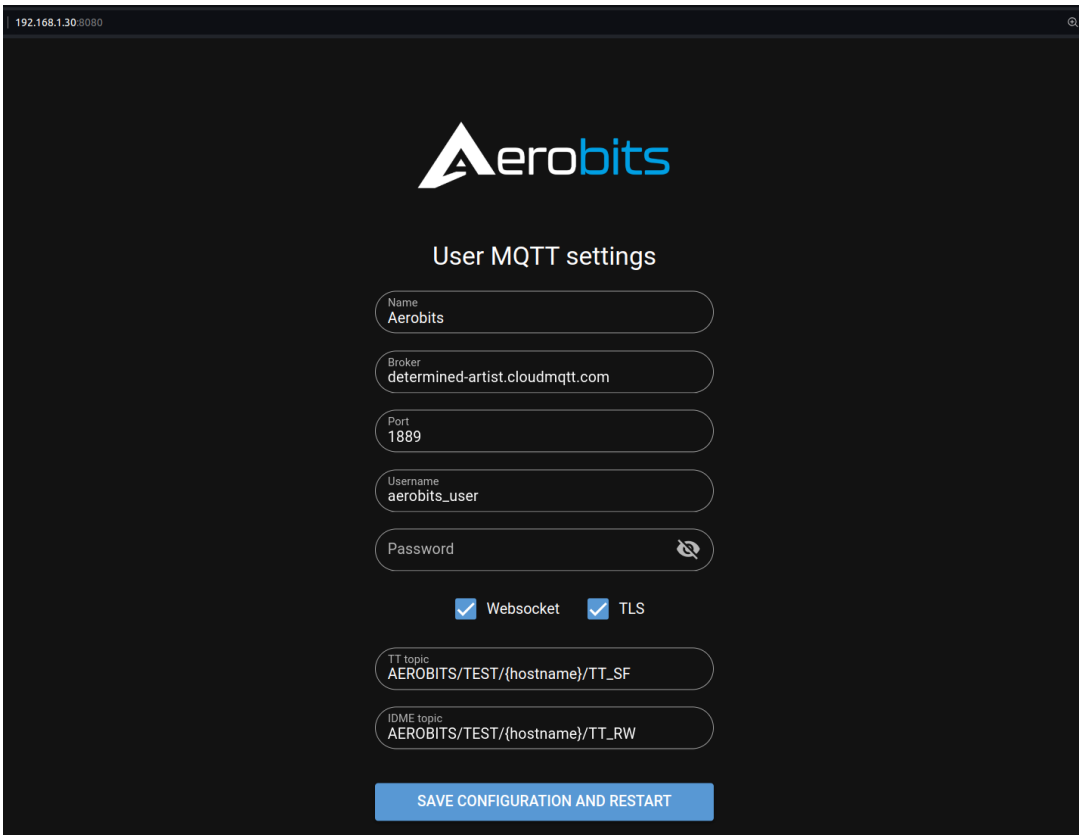
- **IP**: `local device IP`
- **PORT**: `8080`

Example connection shown below:

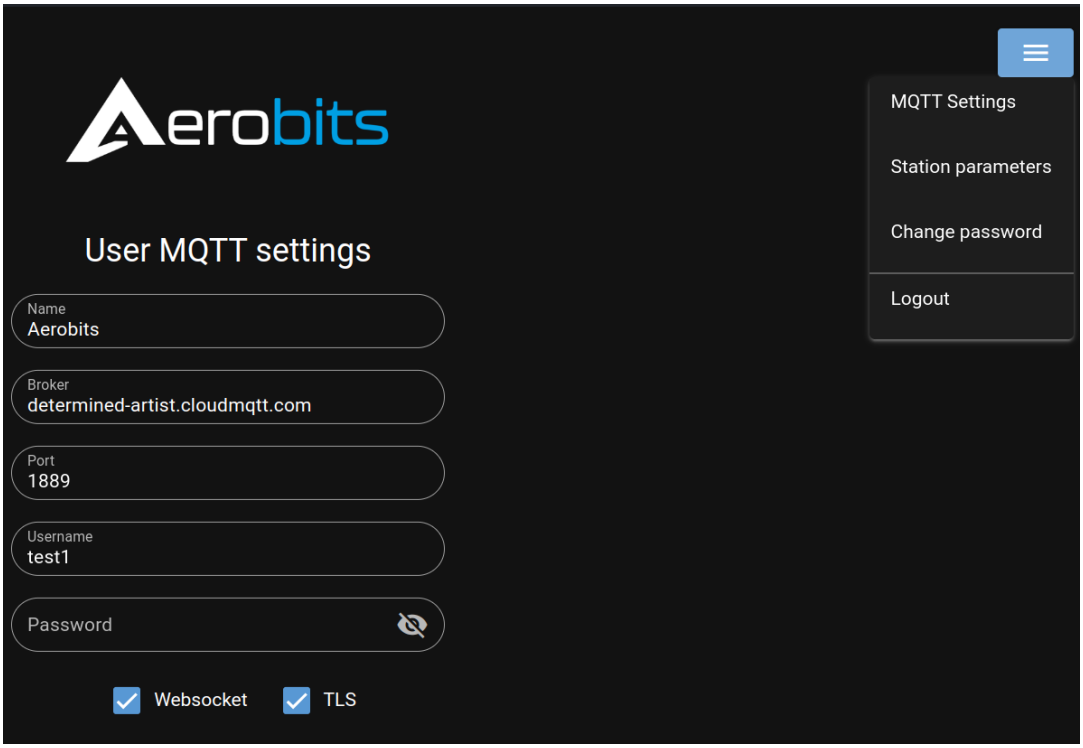
1. Connect to Your device typing IP:PORT in browser. Type password supplied with the device and refresh page after login.

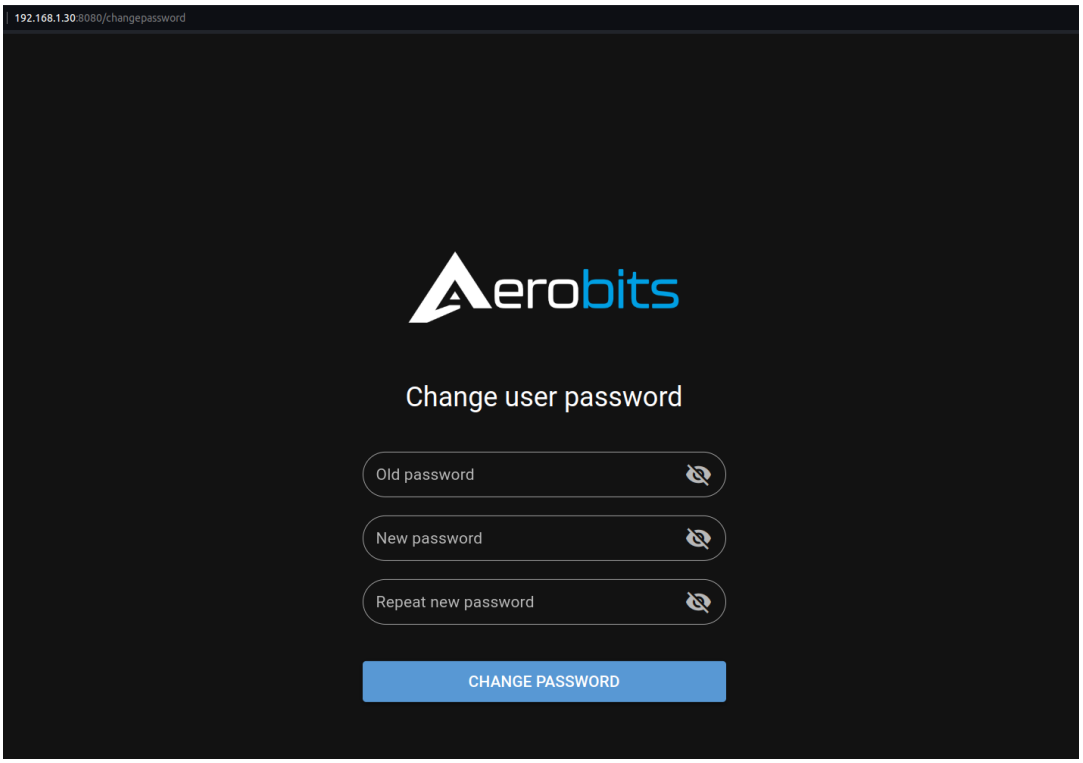


2. Setup Your MQTT connection properties and save configuration.



3. Open menu in the top right corner and change Your custom password and save it.




192.168.1.30:8080/changepassword
Aerobits
Change user password
Old password
New password
Repeat new password
CHANGE PASSWORD

4.3.2 Receiving with PlaneMap

1. install PlaneMap, version at least 0.6.9
2. click **Add Connection** and fill:
   (a) IP:PORT
   (b) MQTT
   (c) login
   (d) password
3. click **Apply and Use**
4. click **Connect**
5. status bar at the bootom should have both **online** and **connected**
6. click **Add Topic**
7. fill topic
8. field **Type** selects frames:
   (a) **AeroJSON:STATUS** is sending data constantly, use it to check if station has connection
   (b) **AeroJSON:GNSS** gives position if it can be received
   (c) **AeroJSON:STATION PARAM** gives sensor reports, every few seconds
9. select **AeroJSON:ADSB** to receive planes
10. **subscribe** to receive data
11. **unsubscribe** to change topic or type
12. click **View Data** to confirm that data is received
13. to show planes switch tabs to **Map**

Please read carefully

Information contained in this document is provided solely in connection with Aerobits products. Aerobits reserves the right to make changes, corrections, modifications or improvements to this document, and to products and services described herein at any time, without notice. All Aerobits products are sold pursuant to our own terms and conditions of sale. Buyers are solely responsible for the choice, selection and use of the Aerobits products and services described herein, and Aerobits assumes no liability whatsoever, related to the choice, selection or use of Aerobits products and services described herein. No license, express or implied, by estoppel or otherwise, to any intellectual property rights is granted under this document. If any part of this document refers to any third party products or services, it shall not be deemed a license granted by Aerobits for use of such third party products or services, or any intellectual property contained therein or considered as a warranty covering use, in any manner whatsoever, of such third party products or services or any intellectual property contained therein.

UNLESS OTHERWISE SET FORTH IN AEROBITS TERMS AND CONDITIONS OF SALE, AEROBITS DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY WITH RESPECT TO USE AND/OR SALE OF AEROBITS PRODUCTS INCLUDING, WITHOUT LIMITATION, IMPLIED WARRANTIES OF MERCHAN-TABILITY, FITNESS FOR A PAR-TICULAR PURPOSE (AND THEIR EQUIVALENTS UNDER THE LAWS OF ANY JURISDICTION), OR INFRINGE-MENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. UNLESS EXPRESSLY APPROVED IN WRITING BY AN AUTHORIZED AEROBITS REPRESENTATIVE, AEROBITS PRODUCTS ARE NOT RECOMMENDED, AUTHORIZED OR WARRANTED FOR USE IN LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS, NOR IN PRODUCTS OR SYSTEMS WHERE FAILURE OR MALFUNCTION MAY RESULT IN PERSONAL INJURY, DEATH, OR SEVERE PROPERTY OR ENVIRONMENTAL DAMAGE.

Information in this document supersedes and replaces all previously supplied information.