2022

skypatrol®

Report Protocol

REVISION: 1.21

Contents

Contents 1
Scope 3
Description 3
SRP Frame Structure 3
Start of Frame 4
Tag 4
Sender ID 4
Sequence Number 4
Timestamp 4
Event 4
Length 5
Payload 5
Item Structure 5
Report Items 6
General (0x00) 7
Mask 7
Product ID 8
GPS Time 9
GPS Position 9
GPS Speed+Heading 9
GPS Altitude 10
GPS Accuracy 10
Cellular Service 10
RSSI 11
Voltages 11
GPIO's 11
Odometer 12
Temperature 12
Battery Gauge 12
Engine Hours 13
Network (0x11) 14
Geofence Type (0x42) 16
Version (0x104) 17
Extended Version (0x105) 18

Component Type 18
Version Type 20
Instance ID 21
Version Length 21
Version 21
File Transfer (0x200) 23
Ignition Status (0x211) 23
External Peripheral (0x311) 24
ACK 25
Revision History 26

Scope

This document describes the flexible SKYPATROL Report Protocol (SRP) supported by the SKYPATROL tracking devices (branded as SKYPATROL). This document should be used as a reference by any developers of backend listener and anyone else that uses SRP as the means of communication with SKYPATROL devices.

Description

SRP is a bidirectional protocol, which allows a device and a server to communicate over an IP (Internet Protocol) link. Each message is "self-contained" in the sense that all the information (such as size) necessary to parse a message is contained within the message itself.

The communication link has the following parameters:

Type: UDP or UDP with ACK
Duplex: Half duplex (Device to Server only) or Full-Duplex
Endianness: Big Endian unless specified otherwise

SRP was designed specifically to allow for event dependent data reporting, so the specific data is determined by the event. The data is organized as a series of Tag-Length-Value (TLV) items, thus allowing for any number of items to be included in each message. The format of each item is derived from its Tag. This protocol is in particular useful when the device has a peripheral that is not changing frequently or is optional, such as OBD reader or Cargo Sensor, and is not part of the regular reporting. It also allows to define special reports that only pertain to specific events.

SRP Frame Structure

The link uses the following frame structure:

```
<start_of_frame><tag><sndr_id><seq#><timestamp><event><length><payload>
```

Where:

<start_of_frame>	1-byte field holding the character 0x7D
<tag>	1-byte field giving information about retransmission (ACK's)
<sndr_id>	8-byte field uniquely identifying the sender
<seq#>	2-byte field indicating the sequence in which the report was generated
<timestamp>	4-byte field indicating the time when the report was generated
<event>	1-byte/2-byte field providing the reason the report was generated
<length>	1-byte/2-byte field indicating the length of the <payload> field
<payload>	variable length field consisting of multiple Items

Note: Any fields designated as 1-byte/2-byte fields are unsigned integers using the following encoding: If the 1st byte is less then 0xE0 (decimal 224) then the field size is 1-byte and the value is the value of the entire byte. If the 1st byte is greater-or-equal to 0xE0 then the field size is 2-bytes and the value is the value of the lsb 13-bits of the 2-byte field (most-significant byte first).

Start of Frame

Start-of-Frame is the character 0x7D (denoting SRP). There is no attempt to escape or remove this character if it appears in the rest of the frame.

Tag

Tag is 1-byte field that gives information about the frame:

Bit 7-4	Reserved
Bit 3-2	provides the LSB 2 bits of the number of times this report has been re-sent due to an ACK receive failure
Bit 1-0	Specifies the acknowledgement expected from the server, if any: 00 – No ACK is expected Any nonzero value means the device expects an ACK

Sender ID

Sender ID is a variable length field which is a unique identifier for the hardware. The first 4 bits of the Sender ID field indicate the nature of the ID, as follows:

Sender ID	
4 bits	15 nibbles
0000	IMEI represented in Binary Coded Decimal. The valid IMEI nibbles are 0-9.
0001	MEID represented in HEX format (the leading nibble is 0). The valid MEID nibbles are 0-F.

Sequence Number

Sequence Number is a 2-byte sequential count of the number of generated reports. This number can be used by the server to detect missing reports. The count will start from 0 on device power-up. The sequence number will overflow back to 0.

Timestamp

Timestamp is a 4-byte field identifying when the report was generated (which may be different from when the report was transmitted). The value of the field is an unsigned integer representing Coordinated Universal Time (UTC) in the following encoding:

<seconds> = <timestamp> % 60
<minutes> = int(<timestamp> / 60) % 60
<hours> = int(<timestamp> / (60*60)) % 24
<days> = 1 + int(<timestamp> / (60*60*24)) % 31
<month> = 1 + int(<timestamp> / (60*60*24*31)) % 12 (where 1 = Jan, 2 = Feb, etc.)
<year> = 2000 + int(<timestamp> / (60*60*24*31*12))

Event

Event if a 1-byte/2-byte field (see the note on 1-byte/2-byte encoding) that identifies the trigger that caused the report. For Event codes refer to AT Command Document.

Length

Length is a 1-byte/2-byte field (see the note on 1-byte/2-byte encoding) that indicates the number of bytes in the <payload>.

Note: If the payload length extends beyond the end of the UDP packet, it is an indication that the received UDP packet may have been unintentionally truncated or corrupted. If the UDP packet contains data beyond the end of the payload, it is an indication that the received UDP packet may have been unintentionally corrupted or concatenated with additional data.

Payload

SRP allows for event dependent data reporting, so that the specific data is determined by the event. The data is organized as a series of Type-Length-Value (TLV) Items, thus allowing for multiple items to be included in each message. The format of each item is derived from its Type and is defined in the Item Structure section. Multiple Items may be concatenated into the same payload.

Item Structure

Each Item has the following TLV based structure:

<type><length><value>

Where:

<type>	1-byte/2-byte field identifying the content and the structure of the <value > field
<length>	1-byte/2-byte field indicating the length of the <value> field
<value>	variable length field, which content is based on the Type of the Item. Report Items section enumerates the defined Types and their corresponding Values.

IMPORTANT NOTES for PARSER IMPLEMENTATION to MAXIMIZE FORWARDS COMPATIBILITY:

From time to time additional Item <type>s will be introduced. The parser should ignore all unrecognized <type>s and use the <length> parameter to find the beginning of the next Item. Even if the <type> is recognized by the parser, the parser should use the <length> parameter to advance to the next Item, rather than relying on counting bytes in the detailed substructure of the Item.

Any Item may be extended in the future without changing the format of the data already defined. Therefore, in order to maintain forward compatibility, the parser should continue to parse the item and simply ignore any extra data.

For any Items that contain a bitmask that describes the item fields and allows to include or exclude them accordingly, the bitmask is always defined starting from the Least Significant Bit. Therefore, any bitmask/field entries beyond those already supported by the parser should be discarded.

Report Items

The collection of Report Items is what gives the reports their flexibility. A report can include any number of Report Items. From time-to-time Report Items will be added. The item list depends on the Event of any given report. Refer to AT Command document for details how to specify which Report Items are included for any given event.

A Report Item typically includes one or more fields that have logical tie-in (e.g. Network Parameters). For convenience, the protocol specifies a number of special Report Items that compound several fields. This allows to use a single Report Item for most of the events and to configure a "default report item".

A few of the Report Items contain a mask field as the first field of their payload. This allows for additional level of flexibility to specify which fields within the Report Item are included in the report.

Table *1* lists the Report Items supported. Grayed out entries are not currently supported, but are planned to be added in the future.

Type (Hex)	Type (Dec)	Value (Content)
0x00	0	**General**: The recommended report for most of the events
0x11	17	**Network**: A maskable collection of cellular network status parameters. Mostly useful for troubleshooting.
0x41	65	**Odometer:** Odometer reading
0x104	260	**Versions:** Versions of the different product s/w and h/w components
0x200	512	**File Transfer Status**
0x211	529	**Ignition Status:** Status of all possible ignition triggers
0x500	1280	**GPS Stream:** sample stream of GPS data (refer to Stream Report)
0x501	1281	**Accelerometer Stream:** sample stream of accelerometer data (refer to Stream Report)
0x1FFF	8191	**Reserved**

Table 1 Items

General (0x00)

This Item contains several data elements that indicate device location and status. It is defined to be the report item that is used for most of the events. To customize the desired data sent, the data elements can be masked according to the Mask field.

`<mask>`	4 bytes
`<fields>`	variable

Mask

Mask is a 32-bit (4-byte) field that specifies the type of data contained in the data field of the report. Only fields which corresponding bit is set are included in the report. Different reports may be programmed to output different data fields. A properly designed listener should never assume that the reports always have a fixed length. It should check the length of the frame and the mask and parse the data fields that it is aware of. New fields will always be added at the end of the data field so the properly designed listener should just ignore the extra fields.

Mask	Bit Mask	Content	Length
Bit 0	0x0000 0001	Product ID	1 byte
Bit 1	0x0000 0002	GPS Time	4 bytes
Bit 2	0x0000 0004	GPS Location [Lon.Lat]	8 bytes
Bit 3	0x0000 0008	GPS Speed+Heading [kmh, degrees]]	3 bytes
Bit 4	0x0000 0010	GPS Altitude [m]	2 bytes
Bit 5	0x0000 0020	GPS Accuracy	2 bytes
Bit 6	0x0000 0040	Main Voltage [mV]	2 bytes
Bit 7	0x0000 0080	Battery Voltage [mV]	2 bytes
Bit 8	0x0000 0100	Aux Voltage [mV]	2 bytes
Bit 9	0x0000 0200	Solar Voltage [mV]	2 bytes
Bit 10	0x0000 0400	Cellular Service	5 bytes
Bit 11	0x0000 0800	RSSI	1 byte
Bit 12	0x0000 1000	GPIO A-D	1 byte
Bit 13	0x0000 2000	GPIO E-H	1 byte
Bit 14	0x0000 4000	Odometer [km]	4 bytes
Bit 15	0x0000 8000	Temperature [0.1°C]	2 bytes
Bit 16	0x0001 0000	Battery Gauge [%, mAh]	3 bytes
Bit 17	0x0002 0000	Engine Hours [sec]	4 bytes
Bit 18	0x0004 0000	Reserved	
Bit 19	0x0008 0000	Reserved	
Bit 20	0x0010 0000	Reserved	
Bit 21	0x0020 0000	Reserved	
Bit 22	0x0040 0000	Reserved	
Bit 23	0x0080 0000	Reserved	
Bit 24	0x0100 0000	Reserved	
Bit 25	0x0200 0000	Reserved	
Bit 26	0x0400 0000	Reserved	
Bit 27	0x0800 0000	Reserved	

Bit 28	0x1000 0000	Reserved	
Bit 29	0x2000 0000	Reserved	
Bit 30	0x4000 0000	Reserved	
Bit 31	0x8000 0000	Reserved	

Table 2 - Report Fields

Default mask value: 0x0000FFFF

Product ID

The **Product ID** is a 1-byte field that indicates the specific product from the SKYPATROL device family thatthe report is sent from. Its objective is to assist a listener that supports multiple products to interpret the report according to the specific product. The table below specifies the product ID values.

Product ID	Product Name	Description
0		Reserved
1	Lynx	Basic LTE Cat-1 Tracker
2	Bobcat-H	Basic HSPA Tracker
3	Puma-C	Trailer CDMA
4	Puma-H	Trailer HSPA
5	Jaguar-H	Trailer w/ Solar HSPA
6	Jaguar-L	Trailer w/ Solar LTE Cat-1
7	Leopard	Basic LTE Cat-M Tracker
8	Lioness	Basic LTE Cat-1 Tracker w/3G fallback
9	Puma-L	Trailer LTE Cat-1 w/3G fallback
10	Panther	2G Tracker
11	Simba-L	Trailer LTE Cat-1 w/3G fallback
12	Wildcat	Basic LTE Cat-M Tracker (BG96 module)
13	Simba-M	Trailer LTE Cat-M Tracker
14	MA60	Slap & Track CAT-M Tracker
15	Nala-L	Trailer w/ Solar LTE Cat-1 w/3G fallback
16	SLP-01	Basic LTE Cat-M Tracker (Nordic MCU)
17	Manx	Basic LTE Cat-M tracker (BG95 module)
18		
19		
20		
21		
22		
23		
24		

Table 3 – Product ID

GPS Time

GPS Time is the Coordinated Universal Time (UTC) associated with the reported device location. If the Location data is current, this timestamp will match the timestamp in the report header. If the device location is old because the current location is unavailable, the timestamp will reflect the time of the last available location. The encoding is the same as the timestamp in the report header, as follows:

<seconds> = <timestamp> % 60
<minutes> = int(<timestamp> / 60) % 60
<hours> = int(<timestamp> / (60*60)) % 24
<days> = 1 + int(<timestamp> / (60*60*24)) % 31
<month> = 1 + int(<timestamp> / (60*60*24*31)) % 12 (where 1 = Jan, 2 = Feb, etc.)
<year> = 2000 + int(<timestamp> / (60*60*24*31*12))

GPS Position

GPS Position is the position reported by the Device GPS subsystem. The format is as follows:

GpsLocation	
Lat	**Lon**
4 bytes	4 bytes

Where:

Lat is a 4-byte representation of the Latitude sent as encodedLat, a big endian unsigned number.
Decoding: `double latitude = ((double)encodedLat / 1000000.0) - 90.0`

Lon is a 4-byte representation of the Latitude sent as encodedLon, a big endian unsigned number.
Decoding: `double longitude = ((double)encodedLon / 1000000.0) - 180.0`

For information the encoding is performed as follows:

```
encodedLat = (uint32)((latitude + 90.0) * 1000000.0)
encodedLon = (uint32)((longitude + 180.0) * 1000000.0)
```

Note: At the time the report is generated, if the device has never fixed a location since power-up, then Time, Lat, and Lon will report all 0's.

GPS Speed+Heading

GpsSpeedHeading	
Speed	**Heading**
1 byte	2 bytes

Speed is an unsigned 1-byte field which indicates the speed of the device in kilometer/hours. The value can be converted into MPH by dividing the reported value by 1.609

Heading is an unsigned 2-byte field which indicates the heading of the device in degrees. Valid values are between 0 and 359.

GPS Altitude

Altitude is a signed 2-byte field which indicates the altitude of the device in meters. The value can be converted into feet by multiplying it by 3.28084. Valid values are between -1000 and +10000.

Altitude
2 bytes

GPS Accuracy

GPS Accuracy is a 2-byte field which gives information about the status and the accuracy of the GPS signal. Its format is as follows:

GPS Accuracy		
Status	**SSkypatrollites**	**HDOP**
4 bits	4 bits	1 byte

Where:
Status is a 4-bit field:

Bit 7	Reserved
Bit 6-4	000: GPS is not locked 001: GPS is locked Other values reserved

SSkypatrollites indicates the number of sSkypatrollites used in the position reports. If the device has never acquireda valid fix, the reported number of sSkypatrollites will be zero.

Bit 3-0	SSkypatrollite count

HDOP is an indication of the quality of the received signal. The reported value must be divided by 10 to obtain the actual HDOP.

Cellular Service

Service is a 1-byte field which gives information about the type of the service (GSM, CDMA, etc.) and whether the device is roaming. Its format is as follows:

Service			
Roaming	**ServiceType**	**MCC**	**MNC**
1 bit	7 bits	2 bytes	2 bytes

Where:
Roaming is a 1-bit field indicating the roaming status:

Bit 7	0: No roaming or no service 1: Roaming

ServiceType is a 7-bit field indicating the cellular service class:

Bit 6-0	0: No service 1: CDMA 1x 2: GPRS 3: HSPA 4: LTE-Cat1 5: LTE-M1 Other values reserved

MCC is the MCC of the carrier. 0xFFFF indicates no service.
MNC is the MNC of the carrier. 0x0 indicates no service.

RSSI

RSSI is a 1-byte field providing a qualitive indication of the RF signal strength seen by the device on a scale 0-31, where 0 indicates a very weak signal, 31 a very strong signal. Value of 99 indicates no signal at the time the report was generated.

RSSI
1 byte

Voltages

The report may contain up to four voltage fields, each of which indicates the power measured on the respective power source at the time the report was generated (in units of 1mV). The power fields can be masked individually. Each power field is 2-byte long. The possible power sources and their respective fields are:
MainPower – Main power, which is typically the voltage of the vehicle battery that powers the device.
BatPower – Internal battery voltage for devices that are self-powered or have a backup battery.
AuxPower – Auxiliary power, which is the voltage of external secondary power source, such as vehicle running lights.
SolSRPower – Voltage supplied by Solar panel for devices that support one.

MainPower	**BatPower**	**AuxPower**	**SolSRPower**
2 bytes	2 bytes	2 bytes	2 bytes

GPIO's

The report contains up to two GPIO fields, **GPIO1, GPIO2.** Each field is a 1-byte field that represent up to 4 GPIO. The GPIO are designated as A, B, C, etc. The precise mapping of the letter codes to pin numbers depends on the hardware. The GPIO fields have the following format:

GPIO1			
D-status	C-status	B-status	A-status
2 bits	2 bits	2 bits	2 bits

GPIO2			
H-status	G-status	F-status	E-status
2 bits	2 bits	2 bits	2 bits

Status: Each X-status field contains the following:

In/Out	Value
1 bit	1 bit

In/Out: 0 for output and bidirectional (including drive hi/lo, open-drain)
1 for input (regardless of high impedance input, pullup resistor, pulldown resistor)

Value:
For input:
0 = input driven low (or external pin is floating and pin is configured to pull low)
1 = input driven high (or external pin is floating and pin is configured to pull high)

For output:
0 = output is driven low or open drain output is floating
1 = output is driven high or open drain output is driving (to ground)

Note: <status> will be '00' if the GPIO is not present in the hardware

Odometer

Odometer is a 4-byte field with the following format:

Odometer
4 bytes

The odometer reports in km.

Temperature

Temperature is a 2-byte field indicating the temperature measured inside the device in the following format:

Temperature
2 bytes

The Temperature is reported in 0.1°C.

Battery Gauge

The Battery Gauge field provides information regarding the charge level of the battery for battery powered devices. The information is provided both as a percentage of the full battery charge and in absolute mAh units. Both numbers are rounded down to the closest integers (e.g. 56.7% is reported as 56%). Battery Gauge has the following format:

Percentage	**Charge**
1 byte	2 bytes

Where:
Percentage – Charge percentage of the battery (relatively to being fully charged).
Charge – Charge level of the battery in mAh.

Engine Hours

EngineHours is a 4-byte field indicating in seconds the total time since “factory reset” that the ignition was on.

EngineHours
4 bytes

Network (0x11)

This Item contains several distinct data elements that indicate cellular network related parameters. It is mostly useful for troubleshooting.

`<mask>` 1 byte
`<fields>` variable

The `<mask>` indicates which fields are present in the `<fields>` field.

Mask	Bit Mask	Content	Bytes
Bit 0	0x01	Service	1
Bit 1	0x02	MCC/MNC	4
Bit 2	0x04	Tower-ID	6
Bit 3	0x08	Centroid	8
Bit 4	0x10	Band	5
Bit 5	0x20	RSSI	1
Bit 6	0x40	RX,TX,EC	6
Bit 7	0x80	Reserved	

Table 4 - Network Mask

Default mask value: 0x7F

Service is a 1-byte field which gives information about the type of the service (GSM, CDMA, etc.) and whether the device is roaming. Its format is as follows:

Service			
Roaming	**ServiceType**	**MCC**	**MNC**
1 bit	7 bits	2 bytes	2 bytes

Where:
Roaming is a 4-bit field indicating the roaming status:

Bit 7	0: No roaming or no service 1: Roaming

ServiceType is a 7-bit field indicating the cellular service class:

Bit 6-0	0: No service 1: CDMA 1x 2: GPRS 3: HSPA 4: LTE-Cat1 5: LTE-M1 Other values reserved

MCC is the MCC of the carrier. 0xFFFF indicates no service.

MNC is the MNC of the carrier. 0xFFFF indicates no service.

RSSI is a 1-byte field providing an indication of the RF signal strength seen by the device on a scale 0-31, where 0 indicates a very weak signal, 31 a very strong signal. Value of 99 indicates no signal at the time the report was generated.

RSSI
1 byte

Tower-ID CDMA:

SID	NID	BID
2 bytes	2 bytes	2 bytes

Tower-ID GSM:

LAC	RNC	CI
2 bytes	2 bytes	2 bytes

Tower-ID LTE:

LAC	CID
2 bytes	4 bytes

Centroid:

Lat	Lon
4 bytes	4 bytes

Band CDMA:

BCx	Chan	PPN
1 byte	2 bytes	2 bytes

BCx is the frequency band code
Chan is the channel number within the frequency band
PPN is the sub-channel

Band GSM:

BCx	Chan	PPN
1 byte	2 bytes	2 bytes

RX/TX/EC-IO

RX	TX	EC
2 bytes	2 bytes	2 bytes

Geofence Type (0x42)

Indicates the number of geo points in a geofence that triggered a Geofence Enter/Exit event.

`<geo_points>` 1 byte

1 point implies a circular Geofence
2 points imply a rectangular Geofence
3 or more points imply a polygon geofence

Version (0x104)

This item contains all the versions pertaining to the device, such as hardware revision, various software components and configuration file of the device.

`<mask>` 1 byte
`<fields>` variable

The `<mask>` indicates which fields are present in the `<fields>` field as follows.

Mask	Bit Mask	Content	Bytes
Bit 0	0x01	HID	3
Bit 1	0x02	O/S (Binary) main version	4
Bit 2	0x04	O/S (Binary) extended version	2
Bit 3	0x08	App	5
Bit 4	0x10	Extender – MCU	4
Bit 5	0x20	Configuration	2
Bit 6	0x40	Reserved	
Bit 7	0x80	Reserved	

Table 5 – Version Mask

Default mask value: 0x3F

Fields that are not applicable to a certain device will be all '0''s. It is recommended to mask out the irrelevant fields.

Mask	**HID**	**O/S main**	**O/S extended**	**App**	**Extender**	**Configuration**
1 byte	3 bytes	4 bytes	2 bytes	5 bytes	4 bytes	2 bytes

HID is a 3-byte value indicating the hardware model and revision of the device. This is internally composed of a 2-byte Model and 1-byte Revision code, MM.R.

O/S main is a 4-byte value indicating the revision of the operating system. This is internally composed of 1-byte Major, Middle, Minor, and Build components X.Y.Z.B.

O/S extended is a 2-byte value indicating the build or the secondary version of the operating system available on some of the devices. This is represented as a single unsigned number.

App is a 5-byte value indicating the version and revision of the Application. This is internally composed of 1-byte Type, Major, Middle, Minor, and Build components T.X.Y.Z.B

Extender is a 4-byte value indicating the revision of the Extender firmware. This is internally composed of a 1-byte Major, Middle, Minor, and Build components X.Y.Z.B. Extender version only applicable to devices with secondary MCU, such as battery powered devices

Configuration is a 2-byte value indicating the version of the Configuration file. The configuration version is defined in the 'devdef' default configuration file. This field has the following format:

Modified	CIN
1 bit	15 bits

Where:
Modified is a 1-bit field that indicates whether an AT command has modified a Default Configuration parameter has been changed.
CIN is a 15-bit field denoting the Config. ID Number which is defined in the Default Configuration file (see +XCIN). The number range is between 0-0x7FFF.

Extended Version (0x105)

This item provides extended information over the Version item. It was primarily designed to accommodate more complex products that consist of any number of components, such as secondary MCU's or external sensors, where each component has its own version (or a number of versions). This item was defined, so it can accommodate various version formats and to be extendable when new components are added over time.

Extended Version item is a series of records. Each record carries information of component version either in string or in numerical format.

`<component1-version><component2-version><component3-version>…` variable

Each record of Component Version consists of the following fields:

Component Type	Version Type	Instance ID (optional)	Version Length (optional)	Version
1 byte	1 byte	2 bytes	2 bytes	Variable

Component Type

Component Type is a 1-byte field indicating the type of the versioned component represented by the record.

The following component types are defined:

Value	Name	Description
0	Reserved	
1	OS (binary) FW	Version of the cellular device firmware (sometimes referred to as binary or as CP), usually provided by either the chipset or the module manufacturer. Usually represented as a string. *Example*: Wildcat FW version provided by Quectel: BG96MAR04A04M1G

2	Application FW	Tracker application FW is provided by AsiSkypatrolco; represented as anumeric four-part version. *Example*: 3.8.10.15
3	Primary (Power) MCU FW	Many trackers have at least one MCU controlling the power of the main module. It is called Primary MCU. Primary MCU FW is provided by AsiSkypatrolco; represented as a numericfour-part version. *Example*: 2.4.0.3
4	Secondary MCU FW	Some tracker designs have additional MCU controlling auxiliary functions of the product like BLE communications, performing MUX functionality, additional computing power, etc. It is called Secondary MCU. Secondary MCU FW is provided by AsiSkypatrolco; represented as a numericfour-part version. *Example*: 1.0.0.2
5	Factory configuration numeric version	Factory configuration numeric version represents a version number embedded in the factory configuration file. It is a 2-byte hexadecimal number. *Example*: 0006 *Note*: version is 0 if no factory configuration file is found or numeric version is not defined in the factory configuration file
6	Factory configuration string version	Factory configuration string version represents a version ID string embedded in the factory configuration file. *Example*: M-KH-VZN *Note*: version is an empty string (with zero length) if no factory configuration file is found or string version is not defined in the factory configuration file.
7	User configuration numeric version	User configuration numeric version represents a version number embedded in the factory configuration file. It is a 2-byte hexadecimal number. *Example*: 0006 *Note*: version is 0 if no user configuration file is found or numeric version is not defined in the user configuration file.

8	User configuration string version	User configuration string version represents a version ID string embedded in the factory configuration file. *Example*: M-KH-VZN *Note*: version is an empty string (with zero length) if no user configuration file is found or string version is not defined in the user configuration file
9	Reserved	
10	Camera BLE MCU FW	FW version of the primary Camera MCU, which is responsible for BLE communication. Since there could be more than one camera attached, this component could be multi-instance component. FW is provided by AsiSkypatrolco; represented as a numeric four-part version. *Example*: 1.2.3.4
11	Camera Secondary MCU FW	FW version of the secondary Camera MCU which is responsible for image processing. Since there could be more than one camera attached, this component could be a multi-instance component. FW is provided by AsiSkypatrolco; represented as a numeric four-part version. *Example*: 1.2.3.4
12	Temperature sensor BLE MCU FW	Temperature sensor MCU FW version. Since there could be more than one temperature sensor attached, this component could be a multi-instance component. FW is provided by AsiSkypatrolco; represented as a numeric four-part version. *Example*: 1.2.3.4
13	Door sensor BLE MCU FW	Door sensor MCU FW version. Since there could be more than one door sensor attached, this component could be a multi-instance component. FW is provided by AsiSkypatrolco; represented as a numeric four-part version. *Example*: 1.2.3.4

Version Type

Version Type is a 1-byte field defining the format of the Version field. This field has the following format:

Multi Instance	Version Format
~~1 bit~~	~~7 bits~~

Where:

Multi Instance is a 1-bit field that indicates whether the item contains multiple records of the same Component Type. If so, each of those records contains an Instance ID field.

Bit 7	0: One instance of this Component Type 1: Multiple instances of this Component Type

Version Format is a 7-bit field indicating the format of the Version field. The following formats are specified:

Bit 6-0	0: Numeric version containing four parts: Major Minor Revision Build 1: Numeric hardware ID containing two parts: Model Revision 2: Numeric version that is represented in 2-byte integer (e.g., configuration file numeric version) 3: String version containing up to 255 characters. When this format is used, the record contains a Version Length field to indicate the actual length of the string.

See Version section below for more detailed description of each of the formats.

Instance ID

Instance ID is an optional 1-byte field that Indicates the instance ID of the Component Type (e.g., peripheral that could be more than one of, such as a temperature sensor). This field is present only if the Multi Instance bit (Bit 7) in the Version Type field is set. Instance ID enumeration starts from 0.

Version Length

Version Length is an optional 1-byte field that Indicates length of the string version. This field is present only if the Version Format of the Version **3** Type field is set to 3 or higher.

Version

The version field can have one of the following formats, according to the identification of the Version Format field:

Version Format 0 – Numeric version containing four parts

In this format, the version is represented by version parts in the following order:

Major version (1 byte)
Minor version (1 byte)
Revision version (1 byte)
Build version (1 byte)

Example: version 1.2.10.20 is represented by the following byte array: 0x01 0x02 0x0A 0x14

Version Format 1 – Numeric hardware ID containing two parts

Hardware ID is represented by 3 bytes divided into two parts: first two bytes represent model number and one byte is for the revision number.

Example: Hardware ID with model 0x3102 and revision 1 is represented by the following byte array: 0x31 0x02 0x01

Version Format 2 – Number (2-byte integer)

Number version is represented by a 2-byte hexadecimal number.

Example: Configuration version 0010 is represented by the following byte array: 0x00 0x10

Version Format 3 – String

String version could contain up to 255 characters without the null terminator and without the surrounding double or single quotes (as indicated in the Version Length field).

Example: version “BL10-45.VZ.200” is represented by the following byte array:
0x42 0x4C 0x31 0x30 0x2D 0x34 0x35 0x2E 0x56 0x5A 0x2E 0x32 0x30 0x30

File Transfer (0x200)

Indicates the status of the last File Transfer operation (via FTP/TFTP/HTTP).

<target> 1 byte
<status> 1 byte
<error_code> 1 byte

Target is a 1-byte field that indicates the file sent via the file transfer protocol. The values and their corresponding targets are as follow:

0 = Reserved
1 = Application
2 = Main MCU (IO controller)
3 = Factory Configuration
4 = User Configuration
5 = Reserved
6 = Cellular Module Firmware
7 = Peripheral Update
8 = Upload to server (not implemented)

Status indicates the status of the file transfer. A '0' indicates a successful file transfer. In this case, the error_code field can be omitted. Any nonzero value indicates a failed file transfer. The value selects a translation table for the error codes that are in the following field.

Error-code specifies the reason of the file transfer failure. The meaning of this value is determined by a translation table that is indicated in the status field. When file transfer status is success, this field is set to '0' and can be omitted.

Ignition Status (0x211)

Indicates the status of all possible ignition triggers.

Ignition Status is a 1-byte field, where every ignition trigger has an associated bit. When bit = 1, the respective ignition trigger is on. Following is the format of the field:

Ignition Status					
Overall ignition status	Reserved	Accelerometer	Reserved	Wired	Virtual
1 bit	3 bits	1 bit	1 bit	1 bit	1 bit

External Peripheral (0x311)

TBD

ACK

The device can be set to request that the server acknowledges the UDP packets. This is accomplished by setting Bit 0 in the Tag field of the SRP header (refer to Tag section). When operating in this mode, a status report will be re-sent over and over until a matching acknowledgement is received from the server.

The server acknowledges the data with the following frame:

<start_of_frame><length><ack_id><tag>[<sndr_id>]

Where:

<start_of_frame>	1-byte field holding the character 0x7E
<length>	2-byte field indicating the length of the remainder of the packet
<ack_type>	1-byte field holding the character 0xE0 (for future extensions)
<tag>	1-byte field providing information about the Ack – bit 6 must match the lsb of the <seq#> field (see Sequence Number) in the frame that is being acknowledged
<sndr_id>	Optional 8-byte field uniquely identifying the sender (currently not supported)

Note: This protocol has no <payload> field.

Example 1: No identification required:

<start_of_frame>	➔ 0x7E
<length>	➔ 0x0002
<ack_type>	➔ 0xE0
<tag>	➔ 0x0F or 0x4F

Revision History

Revision	Date	Description
1.04	02/20/2017	First draft for Skypatrol
1.10	4/19/2021	Changed logos and footer to SKYPATROL and protocol name to SRP TLV 0x200: Modified from TFTP to more general File Transfer. Corrected the file codes. Added Geofence Type (0x42) Corrected field sizes in CS v2 Corrected Temperature TLV to 0.1C units Added Battery Gauge to General TLV Update Product ID table
1.20	6/22/2021	Added Extended Version TLV (0x105)
1.21	8/31/2021	Added Ignition Status TLV (0x211) Added Engine hours to the General report