Communication Protocol

(Between Tracker and Camera)

Author Sam and Yangjie

Revision 1.4

Confidential Material: This document contains information that is proprietary and confidential, reading and copying this document is prohibited without permission from AsiaTelco Technologies Co.

Index

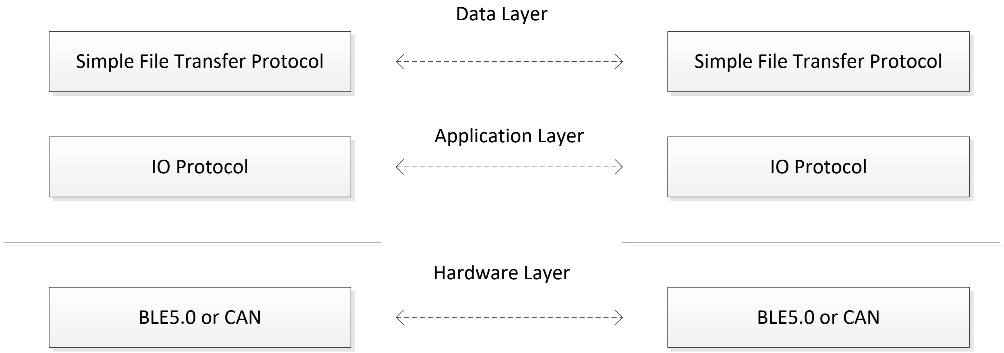
1 Purpose

Camera connects to Tracker via CAN bus or BLE. Camera takes pictures for the Cargo Sensor and send to it. This document defines the protocol of the communication between Camera and Tracker and describes data security over BLE and firmware update of Camera.
Notation list is as the table below:

Notation	
BLE_S	Device works as a BLE slave role. (BLE peripheral)
BLE_M	Device works as a BLE master role. (BLE central)
CAN_S	Device works as CAN server
CAN_C	Device works as CAN client
File_S	The device which sends write request.
File_R	The device which receives write request.
MTU	Maximum Transmission Unit

2 Protocol Stack

The Camera and Tracker are connected based on BLE 5.0/CAN bus. Others are totally same. Let's define the protocol stack of the communication.
Here is the picture of the protocol stack. The same layer in two devices could talk directly and ignore the low layer protocol.



BLE5.0 and CAN works in hardware layer which is common protocol, so let's focus on IO protocol and simple file transfer protocol.

2.1 IO Protocol

IO Protocol is based on the current IO command.
Format:

Sign '$'	Length	Command	Parameters	CR
1 byte	1 byte	1 byte	N bytes	2 bytes

Description:
Sign: $ is the dollar sign (ASCII 0x24).
Length: length of the Parameters.
Command: the type is char.
Parameters : It's a list of optional parameters.
CR : It's the carriage return character (ASCII 0x0D 0x0A).

2.1.1 BLE receive command

Command(ASCII)	**Direction**	**Parameters**	**Notes**

1	Camera ↔Tracker	Simple file transfer protocol	Indicate the command is received from Tracker.

2.1.2 BLE send command

Command(ASCII)	Direction	Parameters	Notes
2	Camera ↔Tracker	Simple file transfer protocol	Indicate the command should be sent to Tracker.

2.2 Simple File Transfer Protocol

Simple file transfer protocol is similar with TFTP protocol. All of the data in this protocol is in little endian. Below is the format of data packet.

opcode	data
2 bytes	N bytes

The generic format is very simple: opcode and data. The size of opcode is always 2 bytes and the size of data depends on the opcode. Below is the definition of the opcode.

Value (Hex)	Value (Dec)	Content
0x02	2	Write Request
0x03	3	Data Pack Body
0x04	4	Retransfer Block
0x64	100	Transfer Request
0x68	104	Request Last Ack
0x69	105	OTA Result

For each opcode, the content of data is different. Here is the definition of the data in different opcode.

2.2.1 Write request(0x02)

Write request is very similar with the write request in TFTP protocol. When Camera has taken a picture and want to send to Tracker, it should send write request to Tracker.

Format:

Filename	Separator	Mode	Separator	Total block
14 bytes	1 byte	6 bytes	1 byte	2 bytes

Description:
Filename: It is a string in ASCII.
Separator: It is always ‘;’.
Mode: For picture, it is always “Binary”
Total block: The type is integer. It is the count of the blocks.

2.2.2 Data packet body(0x03)

Data packet body includes the real data to transfer. Every packet, the data size is always 128 bytes.

Format:

Block Id	Data
2 bytes	128 bytes

Description:
Block Id: The type is integer. It is the id (Block Id and Block Number are the same) of each block. It is from one to maximum block number.

Data: The type is binary and the size is 128 bytes, except the last packet. The size of the last packet depends on the real size of the data.

2.2.3 Retransfer Block(0x04)

Most of the data will be transferred successfully. Only a few of them need to be retransferred. When receiver finds some data packets are missed, it will send retransfer block packet to start the retransfer.

Format:

Number	Block Ids
2 bytes	2*number bytes

Description:

Number: The type is integer. It is the total number of the packets which need to be retransferred.

Block Ids: Each block id occupies 2 bytes, so the total size of block ids is 2*number. Especially, when camera receive it with the number as 1 and block id as 0, it will start to send data packet body. When camera receives it with the number as 1 and block id as total block + 1, all of the data blocks are transferred successfully.

2.2.4 Transfer Request(0x64)

When a picture is requested to be taken, a transfer request packet will be sent to camera. The packet includes the parameters of the picture.

Format:

Image Num	Size Level	Trigger Type	White Balance	Mac	Time
1 byte	1 byte	1 byte	1 byte	6 bytes	7 bytes

Description:

Image Num: The type is integer. It is the number of the picture should be taken.

Size Level: The type is integer. It is the quality of the picture. 1: low quality. 2: mid quality. 3: high quality.

Trigger Type: The type is integer. It is the reason of the picture. For cargo sensor, depend on the requirement.

White Balance: The type is integer. It is the white balance of the picture based on the current environment. 0: auto. 1: sunny. 2: cloudy. 3: office. 4: home.

Mac: The type is raw binary. It is the Mac address of the device, who sends the transfer request to camera.

Time: The type is date and time in hex. Its format is yyyymmddhhmmss. There are 14 digits in the format. The size of the time is 7 bytes. Each byte has two digits which mean two digits in the format.

2.2.5 Request Last Ack(0x68)

Request last ack is sent from the camera to check the picture is sent successfully or not. This packet doesn't have any data.

2.2.6 OTA Result(0x69)

Camera will feedback the firmware upgrade result to Tracker.

Format:

OTA Result Value
2 bytes

Description:

Packet ID: The type is integer, range: 0x0000 – 0x0002. 0x0000: OTA success, 0x0001: OTA timeout error, 0x0002: OTA upgrade error.

3 Working procedure

Tracker works as a central device and Camera works as a peripheral device. Tracker and Camera should be connected in pair and after that they can communicate with each other(If they are connected with each other via BLE). The working procedure includes two parts, connection established and file/picture transfer.

3.1 BLE Connection Establishment

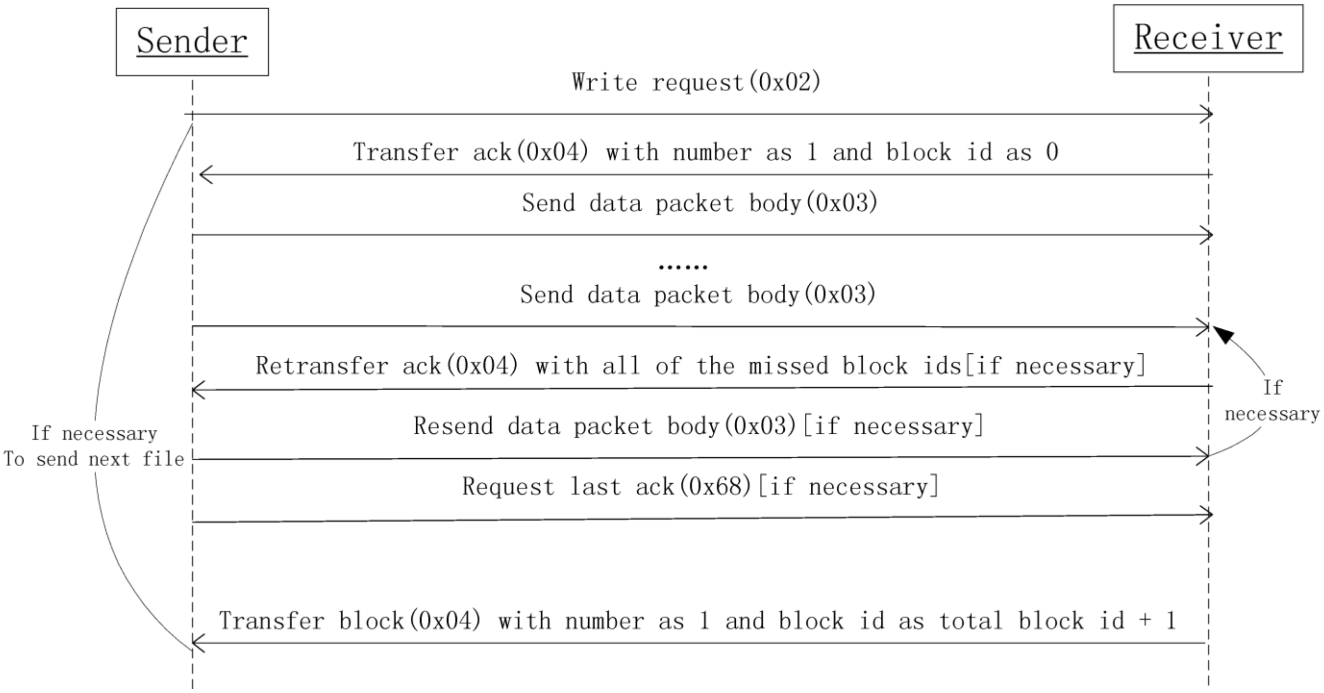
Please refer to document "Tracker Camera Pairing and Unpairing".

3.2 File transfer

For file transfer, the simple file transfer protocol will be followed. Both Camera and Tracker can work as a File Sender or File Receiver. The procedure is described in the following steps.
1, File Sender sends write request(0x02) to File Receiver, which includes the file name, the number of the total blocks.
4, When File Receiver receives it, create a file to save it and save the number of the total blocks.
5, File Receiver sends a retransfer block(0x04) with number as 1 and block id as 0.
6, When File Sender receives it, start to send all of the data packet body(0x03) packets to File Receiver one by one continuously.
7, When File Receiver receives the data packet body(0x03), it will check the block id of each packet. Because the data packets are sent one by one, the block id should be continuously, so the expected block id could be calculated. If it is not the expected block id, File Receiver will send retransfer block(0x04) with all of the missed block ids.
8, When File Sender receive it, it will send all of the missed blocks to File Receiver.
9, When File Receiver receives all of the data blocks, it will send retransfer block(0x04) with number as 1 and block id as total block id + 1.
10, When File Sender receives it, the file transfer is finished successfully. Then File Receiver will send the next file if required. If there is not any packet received after 2 seconds, when the last data packet body(0x03) is sent, File Sender will send request last ack(0x68) to File Receiver.
11, When File Receiver receives it, it will check whether all of the data blocks are received successfully or not, if yes, it will send retransfer block(0x04) with number as 1 and block id as total block id + 1 to File Sender. If no, it will send retransfer block(0x04) with all of the missed block ids to File Sender.
12, For File Sender, if it sends request last ack(0x68) but doesn't receive the retransfer block(0x04) with number as 1 and block id as total block id + 1, it will repeat 7 times to make sure the whole procedure could be finished successfully.

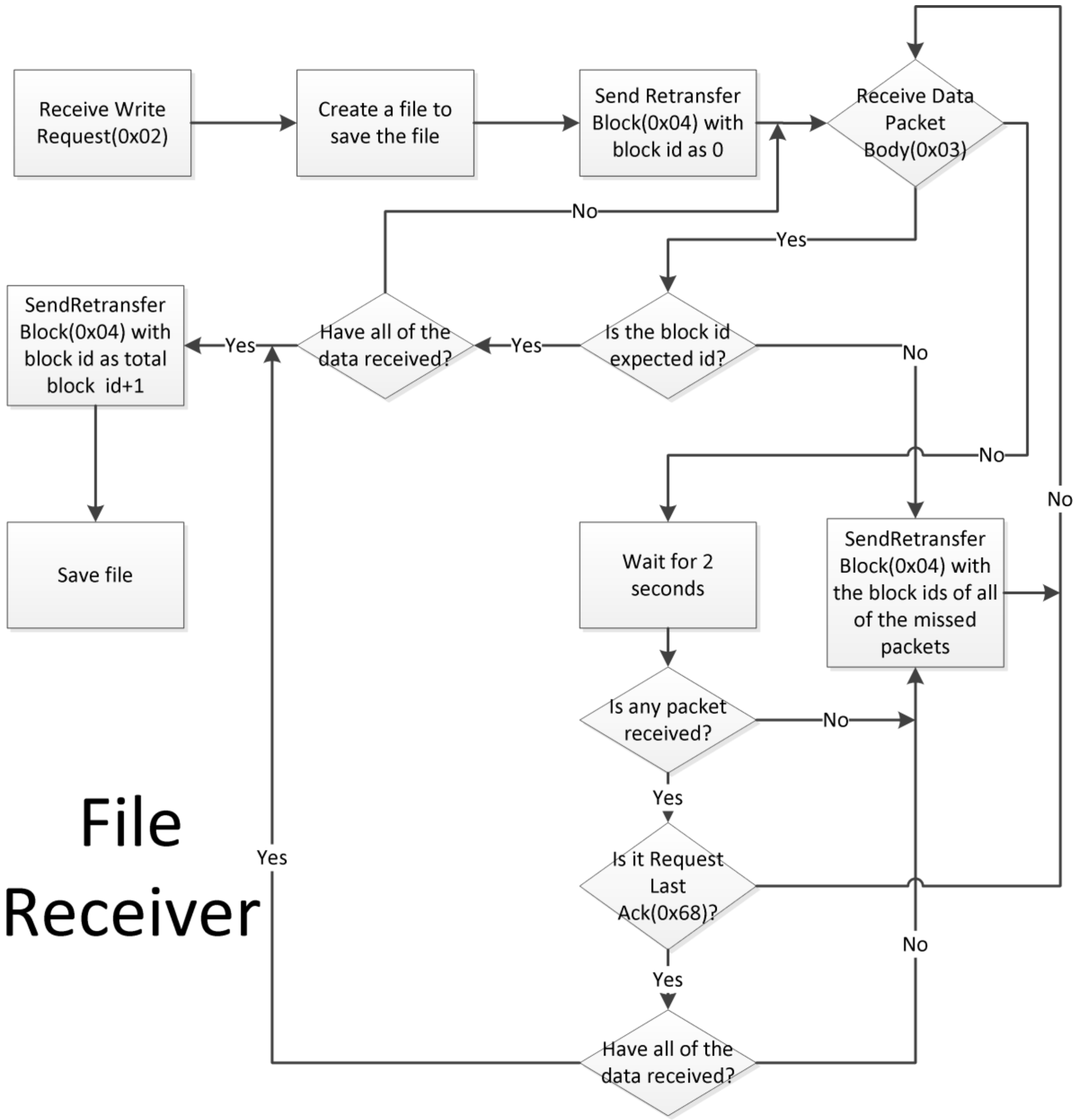
3.2.1 Sequence diagram

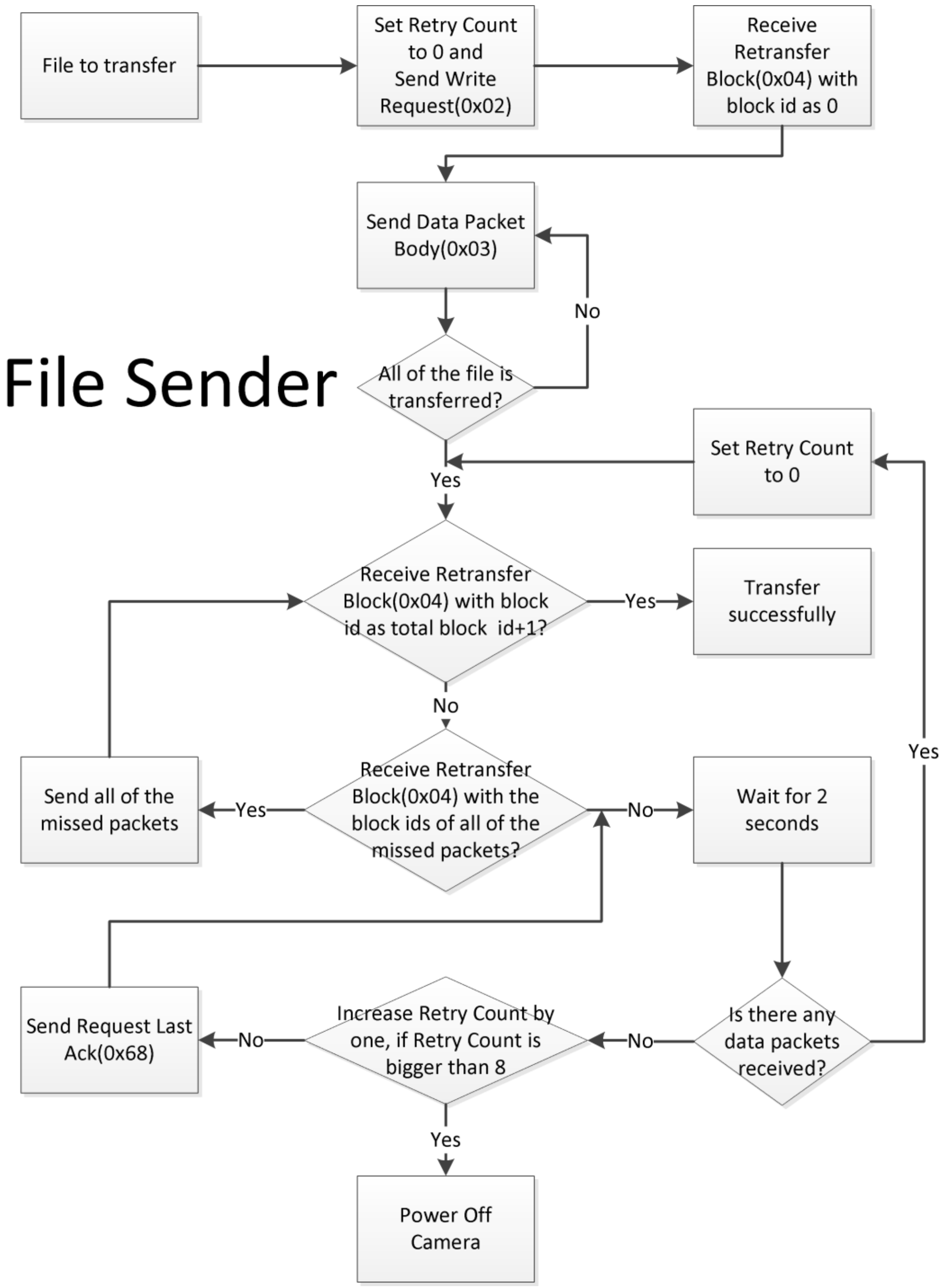
Below is the sequence diagram to show the whole procedure.


Sender
Receiver
Write request(0x02)
Transfer ack(0x04) with number as 1 and block id as 0
Send data packet body(0x03)
……
Send data packet body(0x03)
Retransfer ack(0x04) with all of the missed block ids[if necessary]
If necessary
Resend data packet body(0x03)[if necessary]
If necessary
To send next file
Request last ack(0x68)[if necessary]
Transfer block(0x04) with number as 1 and block id as total block id + 1

3.2.2 Work flow diagram

As mentioned above, here are the work flow diagrams for File Sender and File Receiver, when they are transferring files.


Receive Write Request(0x02)
Create a file to save the file
Send Retransfer Block(0x04) with block id as 0
Receive Data Packet Body(0x03)
No
Yes
SendRetransfer Block(0x04) with block id as total block id+1
Yes
Have all of the data received?
Yes
Is the block id expected id?
No
No
No
Save file
Wait for 2 seconds
SendRetransfer Block(0x04) with the block ids of all of the missed packets
Is any packet received?
No
Yes
Is it Request Last Ack(0x68)?
Yes
No
Yes
Have all of the data received?
File Receiver


File to transfer
Set Retry Count to 0 and Send Write Request(0x02)
Receive Retransfer Block(0x04) with block id as 0
Send Data Packet Body(0x03)
No
File Sender
All of the file is transferred?
Set Retry Count to 0
Yes
Receive Retransfer Block(0x04) with block id as total block id+1?
Yes
Transfer successfully
No
Yes
Send all of the missed packets
Yes
Receive Retransfer Block(0x04) with the block ids of all of the missed packets?
No
Wait for 2 seconds
Send Request Last Ack(0x68)
No
Increase Retry Count by one, if Retry Count is bigger than 8
No
Is there any data packets received?
Yes
Power Off Camera

3.3 Picture transfer

Picture transfer is based on file transfer, but Tracker need to start the procedure with some parameters. For picture transfer, the simple file transfer protocol will be followed. The procedure is described in the following steps.

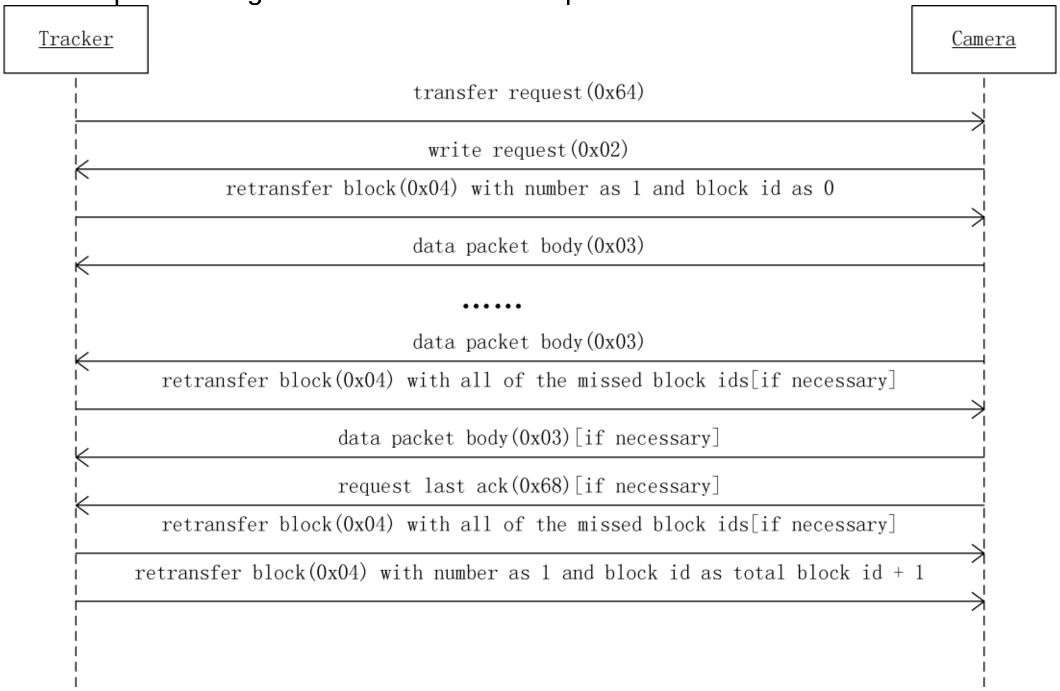
1, Tracker (Cargo sensor) sends transfer request(0x64) to Camera, which includes all of the parameters of the picture.

2, When Camera receives it, take a picture as required.

3, Camera sends write request(0x02) to Tracker (Cargo sensor), which includes the file name, the number of the total blocks.
4, When Tracker (Cargo sensor) receives it, create a file to save it and save the number of the total blocks.
5, Tracker (Cargo sensor) sends a retransfer block(0x04) with number as 1 and block id as 0.
6, When Camera receives it, start to send all of the data packet body(0x03) packets to Tracker (Cargo sensor) one by one continuously.
7, When Tracker (Cargo sensor) receives the data packet body(0x03), it will check the block id of each packet. Because the data packets are sent one by one, the block id should be continuously, so the expected block id could be calculated. If it is not the expected block id, Tracker (Cargo sensor) will send retransfer block(0x04) with all of the missed block ids.
8, When Camera receive it, it will send all of the missed blocks to Tracker (Cargo sensor).
9, When Tracker (Cargo sensor) receives all of the data blocks, it will send retransfer block(0x04) with number as 1 and block id as total block id + 1.
10, When Camera receives it, the picture transfer is finished successfully. Camera of Camera will be off. If there is not any packet received after 2 seconds, when the last data packet body(0x03) is sent, Camera will send request last ack(0x68) to Tracker (Cargo sensor).
11, When Tracker (Cargo sensor) receives it, it will check whether all of the data blocks are received successfully or not? If yes, it will send retransfer block(0x04) with number as 1 and block id as total block id + 1 to Camera. If no, it will send retransfer block(0x04) with all of the missed block ids to Camera.
12, For Camera, if it sends request last ack(0x68) but doesn't receive the retransfer block(0x04) with number as 1 and block id as total block id + 1, it will repeat 7 times to make sure the whole procedure could be finished successfully.

3.3.1 Sequence diagram

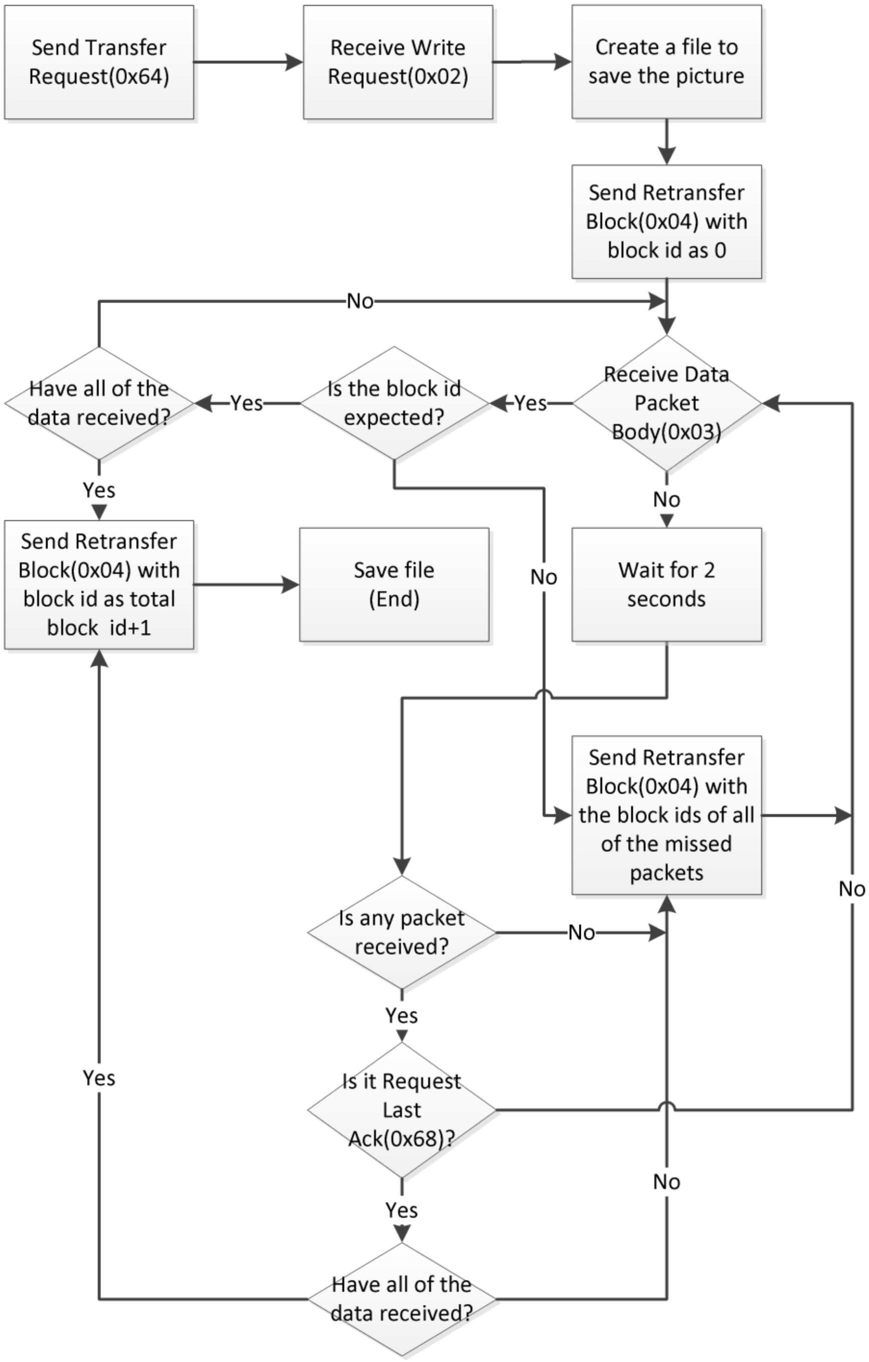
Below is the sequence diagram to show the whole procedure.

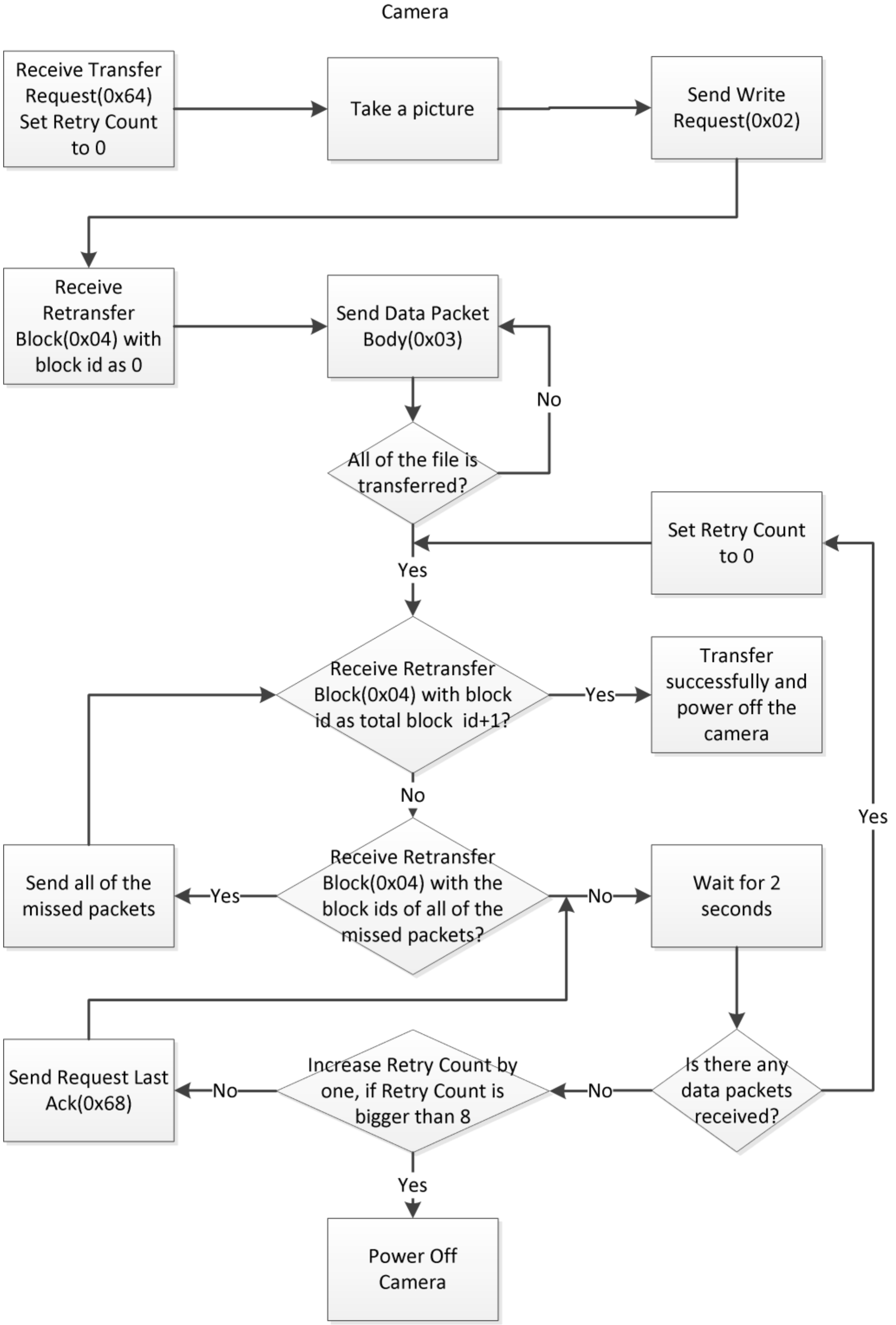

Tracker
Camera
transfer request(0x64)
write request(0x02)
retransfer block(0x04) with number as 1 and block id as 0
data packet body(0x03)
......
data packet body(0x03)
retransfer block(0x04) with all of the missed block ids[if necessary]
data packet body(0x03)[if necessary]
request last ack(0x68)[if necessary]
retransfer block(0x04) with all of the missed block ids[if necessary]
retransfer block(0x04) with number as 1 and block id as total block id + 1

3.3.2 Work flow diagram

As aforementioned, here are the work flow diagrams for Tracker (Cargo sensor) and Camera, when they are transferring pictures.

Tracker (Cargo Sensor)


Send Transfer Request(0x64)
Receive Write Request(0x02)
Create a file to save the picture
Send Retransfer Block(0x04) with block id as 0
No
Have all of the data received?
Yes
Is the block id expected?
Yes
Receive Data Packet Body(0x03)
Yes
No
Send Retransfer Block(0x04) with block id as total block id+1
Save file (End)
No
Wait for 2 seconds
Send Retransfer Block(0x04) with the block ids of all of the missed packets
No
Is any packet received?
No
Yes
Yes
Is it Request Last Ack(0x68)?
No
Yes
Have all of the data received?


Camera
Receive Transfer Request(0x64) Set Retry Count to 0
Take a picture
Send Write Request(0x02)
Receive Retransfer Block(0x04) with block id as 0
Send Data Packet Body(0x03)
No
All of the file is transferred?
Set Retry Count to 0
Yes
Receive Retransfer Block(0x04) with block id as total block id+1?
Yes
Transfer successfully and power off the camera
No
Yes
Send all of the missed packets
Yes
Receive Retransfer Block(0x04) with the block ids of all of the missed packets?
No
Wait for 2 seconds
Send Request Last Ack(0x68)
No
Increase Retry Count by one, if Retry Count is bigger than 8
No
Is there any data packets received?
Yes
Power Off Camera

4 Communication Security

When Camera is connected with Tracker via BLE, there is a risk that BLE data can be snifferred over the air. Thus, BLE data should be encrypted. If Camera is paired with Tracker, the BLE data over the air is encrypted (Following BLE standards). For more information refer to document “Tracker Camera Pairing and Unpairing”.

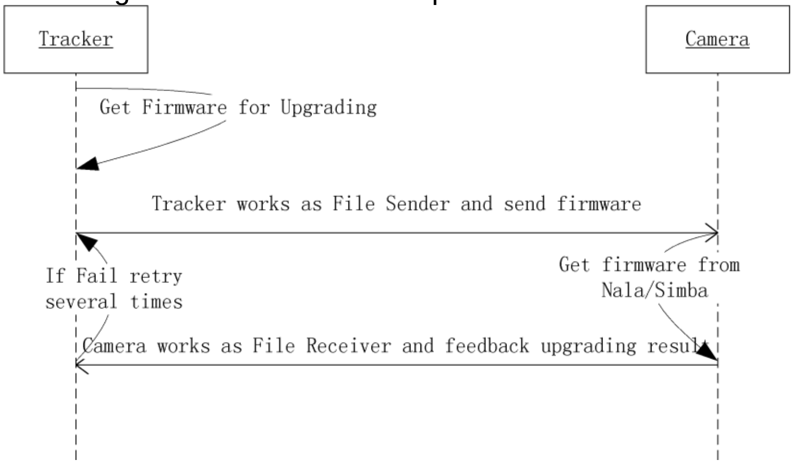
5 Firmware Upgrade

When upgrading Camera firmware, Camera works as File Receiver and Tracker works as File Sender. The procedure is described in the following steps:

1. Tracker get firmware from Network or other sources.
2. Tracker works as File Sender, Camera works as File Receiver, Tracker send the firmware to Camera.
3. If firmware transmission complete, Camera start internal upgrading procedure. If upgrading procedure success feedback OTA Upgrade Success to Tracker, otherwise feedback OTA Upgrade Error.
4. If firmware transmission fail, Camera feedback OTA Timeout Error to Tracker.
5. If upgrading not success, Tracker will retry several times.

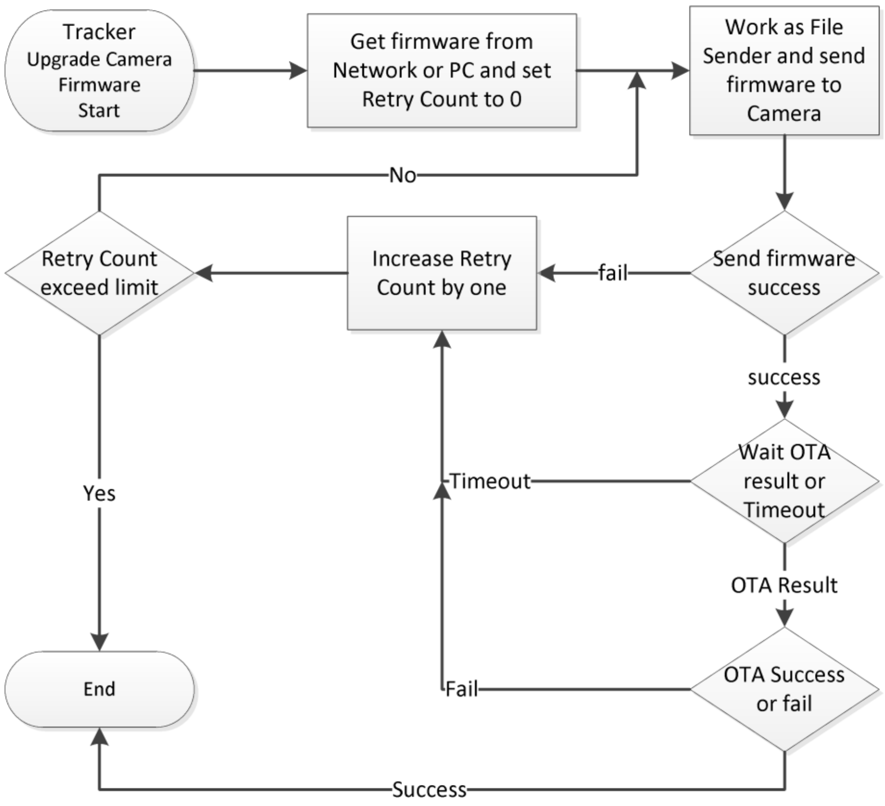
5.1.1 Sequence diagram

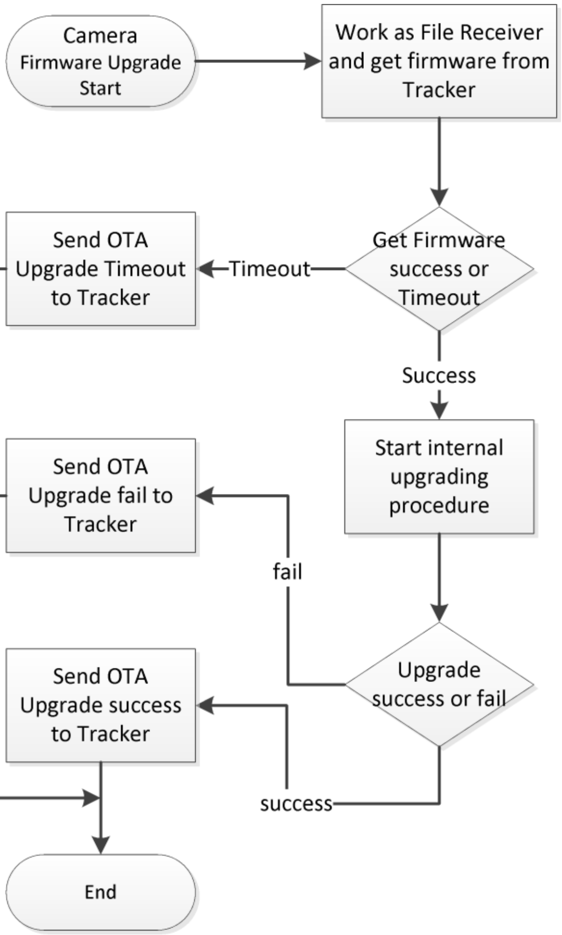
Below is the sequence diagram to show the whole procedure.


Tracker
Camera
Get Firmware for Upgrading
Tracker works as File Sender and send firmware
If Fail retry
several times
Get firmware from
Nala/Simba
Camera works as File Receiver and feedback upgrading result

5.1.2 Work flow diagram

As aforementioned, here are the work flow diagrams for Camera firmware upgrading.


Tracker Upgrade Camera Firmware Start
Get firmware from Network or PC and set Retry Count to 0
Work as File Sender and send firmware to Camera
No
Retry Count exceed limit
Increase Retry Count by one
fail
Send firmware success
success
Yes
Timeout
Wait OTA result or Timeout
OTA Result
End
Fail
OTA Success or fail
Success


Camera Firmware Upgrade Start
Work as File Receiver and get firmware from Tracker
Get Firmware success or Timeout
Timeout
Send OTA Upgrade Timeout to Tracker
Success
Start internal upgrading procedure
fail
Send OTA Upgrade fail to Tracker
Upgrade success or fail
success
Send OTA Upgrade success to Tracker
End

6 Change History

Revision	Author	Date	Change Notes
1.0	Sam	2020/10/11	Draft
1.1	Sam	2020/10/12	Add work flow diagrams for connection and picture transfer
1.2	Yangjie	2021/04/15	Add communication security. Add firmware upgrading.
1.3	Yangjie	2021/04/16	Rename some items.
1.4	Yangjie	2021/04/26	Fix some errors in file transfer/image transfer flowchart.