FUOTA Process Summary Technical Recommendation TR002 v1.0.0

NOTICE OF USE AND DISCLOSURE

FUOTA Process Summary Technical Recommendation

Authored by the FUOTA Working Group of the LoRa Alliance Technical Committee

Technical Committee Chairs:
T.KRAMP (Semtech), A.YEGIN (Actility)

Working Group Chairs:
J.CATALANO (Kerlink), N.SORNIN (Semtech)

Editor:
J.CATALANO (Kerlink)

Contributors:
J.CATALANO (Kerlink), J-P.COUPIGNY (STMicroelectronics), M.KUYPER (TrackNet), N.SORNIN (Semtech), A.YEGIN (Actility)

Version: 1.0.0
Date: January 30, 2019
Status: FINAL RELEASE

Contents

Tables

Figures

1 Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

The octet order over the air for all multi-octet fields is little endian (least significant byte is sent first).

2 Introduction

This document summarizes all the components and process steps involved in an end-device Firmware Update Over-the-Air on top of the LoRaWAN protocol.

The firmware update over the air is not part of the MAC layer but runs at the application layer, as such, all processes described in this document are LoRaWAN protocol version agnostic.

3 Package Identifier List

This section lists the current defined FUOTA packages, their identifier (Package ID), their allocated FPort or example Fport, their status, version and a link to the latest version of specification describing that package.

Package ID	Allocated FPort	Proposed FPort	Package version	Name	Version	Document link	Status	Release Date
0	**225**		1DRAFT	Multi Package Access Protocol over LoRaWAN	RC1	Multi_Package_Access_rc1.docx	Release candidate	
1		202	1	LoRaWAN Application Layer Clock Synchronization	v1.0.0	LoRaWAN Application Layer Clock Synchronization Specification v1.0.0	Published	September 10, 2018
2		200	1	LoRaWAN Remote Multicast Setup Specification	v1.0.0	LoRaWAN Remote Multicast Setup Specification v1.0.0	Published	September 10, 2018
3		201	1	LoRaWAN Fragmented Data Block Transport	v1.0.0	LoRaWAN Fragmented Data Block Transport Specification v1.0.0	Published	September 10, 2018
4		203	1DRAFT	LoRaWAN Firmware Management Protocol	RC1	Firmware_Management_Protocol_rc1.docx	Release candidate	

Table 1: Package Identifier List

4 FUOTA Process Steps

Figure 1 depicts the FUOTA architecture used in this document. Interfaces with dotted lines are outside the scope of the LoRa Alliance. Interfaces with straight lines are handled by the LoRa Alliance specifications.

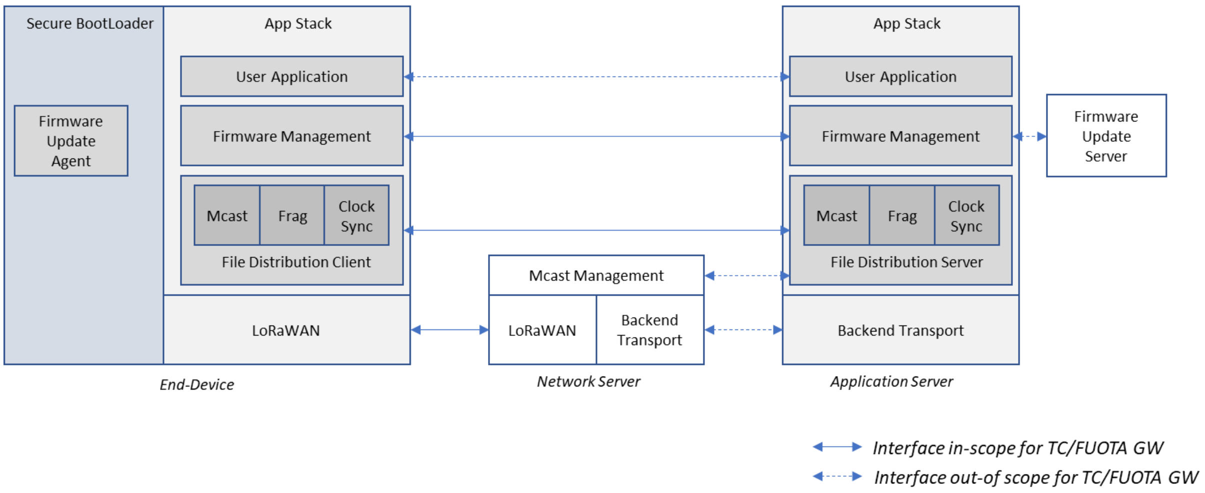

Figure 1: FUOTA architecture

Notations:

FUS	Firmware Update Server. A server tasked to generate the firmware update image to be used on the end-device and the list of end-devices to be updated. FUS interfaces with the Firmware Management Module on the Application Server side using an out of band mechanism.
FUA	Firmware Update Agent. This is the counter-part of the FUS on the End-device side.
FDS	File Distribution Server. An application server specifically tasked to deliver the firmware update image to end-devices. This server operates at the application layer (in parallel or on top of the AS) on a specific set of ports.
FDC	File Distribution Client. This is the counter-part of the FDS on the End-device side.
NS	Network Server
Dev	End-device

Table 2: Process Summary Notations

The following table describes the Firmware Update Over-the-Air process, listing the sequence of actions and their actor to achieve FUOTA.

	Actor	Action
1	FDS	Gather the identifier of all end-devices that will be updated. This step can be bypassed if unicast is used, or if a multicast group already exists that contains the end-devices to be updated. The multicast group may contain more end-devices than the subset of end-devices to be updated. If a delta-firmware image is used, all end-devices selected must currently be running the exact same firmware image.
2	FUS	Creates the new compressed firmware image or delta-image. Select fragmentation parameters (number of fragments, redundancy, erasure code). The image contains a header (at the minimum describing the target end-devices hardware version, current firmware version, image CRC,

		compression mechanism used by the image, etc.) and a digital signature using a private/public key scheme (used to authenticate the image). The image is signed using the FUS private key. Create the list of fragments (the fragment file).
3	FDS-NS	Negotiate with the NS a Class C or Class B distribution window. Parameters include the list of end-devices, the size of the fragment file to send, time criticality, coding redundancy, etc.
4	FDS	If necessary, configure the multicast group using applicative unicast downlink for all end-devices to be updated. (multicast address to use, key material) [McGroupSetupReq/Ans]
5	FDS	Configure Class C or Class B rendezvous using applicative unicast downlink for all end-devices to be updated. [McClassCSessionReq/Ans]. In the background the end-device MUST synchronize its clock to the network's clock because absolute time is used to define the session start.
6	FDS	Setup the fragmentation session for all end-devices to be updated. The fragmentation session setup command contains a freely assigned descriptor such that end-devices, in case of a FUOTA broadcast, can autonomously check that the image is applicable to them and knows where to store it. [FragSessionSetupReq/Ans]
7	FDS-NS	Send fragmented file to the NS.
8	NS-Dev	NS broadcasts (or unicasts) the fragment file to end-devices to be updated.
9	Dev	As soon as an end-device has received enough fragments, it reconstructs the binary image (BLOB).
10	Dev	End-device checks the digital signature of the image and authenticates the sender using the FUS public key stored in all end-devices. This step also serves as an integrity test of the firmware upgrade image.
11	Dev	End-device checks the image header: 1. Is the image compatible with the end-device's hardware? 2. Is the image compatible with the firmware version currently running on the end-device (the firmware version may be a CRC of the code)?
12	Dev	The end-device's application marks the firmware image as "ready". This means that the image will be installed by the bootloader at the next reset. The application may decide to reboot immediately, or may differ the reboot to a later time, or wait for a firmware management command to reboot.
13	Dev	The end-device reboots.
14	Dev	Bootloader checks the availability of a firmware upgrade image and founds one. Bootloader checks again the CRC of the image, un-compresses the image or delta-image if required. This decompression might happen directly in-place, meaning the decompressed image overwrites the previous firmware image. Or the decompression might happen in a currently unused memory space for end-devices supporting dual-firmware image. If the bootloader overwrites the firmware code memory space, then this update is performed transactionally. ("transactionally" means that flash memory is updated page per page, at each step the content of the page written is verified and the address of the next page to update is written to NVM. This process guarantees that even if the end-device crashes or power is interrupted during the update process, the process will resume where it was left at the next reset). Bootloader marks the firmware upgrade image as installed.

15	Dev	End-device reboots. Bootloader checks availability of a firmware upgrade image and finds none, therefore start the newly updated application firmware. If the end-device is an OTAA end-device, it goes to Join mode.
16	Dev-FUS	End-device optionally sends application uplinks containing the new firmware version currently running and time since reboot until acknowledged by the AS (or FUS). (This is part of a device management package yet to be defined)
17	FUS	Update Server collects the firmware versions of all end-devices and mark as updated the ones successfully updated.

Table 3: Process Summary

5 Glossary

ABP	Activation by Personalization
AS	Application Server
BLOB	Binary Large Object
CRC	Cyclic Redundancy Check
FUOTA	Firmware Update Over-The-Air
NS	Network Server
NVM	Non-Volatile Memory
OTAA	Over-The-Air Activation
TR	Technical Recommendation
US	Update Server

6 Bibliography

6.1 References

[TS001 v1.0.2]: LoRaWAN^{TM} 1.0.2 Specification, LoRa Alliance, July 2016

[TS001 v1.1]: LoRaWAN^{TM} 1.1 Specification, LoRa Alliance, October 11, 2017

[TS003 v1.0.0] LoRaWAN Application Layer Clock Synchronization Specification, LoRa Alliance, September 10, 2018

[TS004 v1.0.0]: LoRaWAN Fragmented Data Block Transport Specification, LoRa Alliance, September 10, 2018

[TS005 v1.0.0]: LoRaWAN Remote Multicast Setup Specification, LoRa Alliance, September 10, 2018