GTI
Best Practice for FOTA of NB-IoT Device

Best Practice for FOTA of NB-IoT Device

Version:	1.0
Deliverable Type	**□ Procedural Document** **√Working Document**
Confidential Level	**√Open to GTI Operator Members** **√Open to GTI Partners** **√ Open to Public**
Working Group	
Task Force	
Contributors	CMCC, Qualcomm, Sprint
Editors	
Last Edit Date	06-10-2019
Approval Date	DD-MM-YYYY

Document History

Date	Meeting #	Version #	Revision Contents
2019-01-15		0.1	Skeleton Draft
2019-05-22		0.2	Initial Draft
2019-06-10		1.0	Modify typo errors

Table of Contents

1 Executive Summary

Mobile IoT market is growing rapidly [1]. The Internet of Things devices are gradually penetrating into all aspects of life, and a huge amount of devices are being connected. The large-scale deployment of IoT devices ask for firmware upgrade of FOTA (Firmware Over-The-Air).

Firmware OTA update is essential for large scale IoT deployment. For IoT devices, firmware update is not limited to the “narrow-defined” firmware update, it covers the functions set the IoT device providing in a sense. With the IoT devices continue to become mature, the traditional method to update the devices which would require engineers on site is not feasible for IoT. Thus, if FOTA couldn’t get supported, the critical security patches, bug fixes as well as the latest product features will not be able to deliver to geographically dispersed devises. The IoT network/device would not be able to evolve and be improved continuously, and more important, the IoT network would suffer from severe security vulnerability.

FOTA would become an integral part of the Internet of Things system to turn IoT as a new kind of service for end users. Manufacturers can use FOTA to add new functions, or repair product defects, and bring new technical features to customers after the physical devices are rapidly delivered products to the market.

OTA update is being widely adopted in smart phone devices, large OEMs with strong technical strength usually to deploy FOTA platforms by themselves. However, the IoT device provider’s expertise more focus in the device functions in a physical world, while lack of experiences in information and communication field. It is hard for each IoT devices producers to manage the device remotely. IoT platform operators have to provide the common FOTA capability, though the FOTA is an important enabler to promote large-scale deployment of the Internet of Things.

On the other hand, compared with smart devices such as mobile phones, FOTA upgrade for Internet of Things devices should consider more aspects, including limited storage space, low date rate for image transmission, power consumption, firmware package segmentation and etc.

The white paper analyzes the FOTA requirements of NB-IoT devices, the challenges and chances for FOTA based on the NB-IoT network, and proposes an end-to-end FOTA solution and reference implementation of NB-IoT devices based on LwM2M, Including the architecture and implementation of incremental image generation, firmware image download process, breakpoint continuation, state machine. Particularly, unlike the existing smart devices on use the initiative to query the upgrade package and pull the upgrade package, we chose “PUSH” method for NB-IoT module considering NB-IoT device usually resource constrained devices. Finally, we share an example based on this solution and summarize some important ideas which we have discovered.

2 Abbreviations

Abbreviation	**Explanation**
M2M	Machine to Machine (Communication)
NB-IoT	Narrow Band Internet of Thing
CoAP	The Constrained Application Protocol
UDP	User Datagram Protocol
UE	User Equipment
IPSO	Internet Protocol for Smart Objects
LwM2M	Lightweight Machine to Machine
API	Application Programming Interface
URI	Uniform Resource Identifier
FOTA	Firmware Over The Air
C-IoT	Cellar Internet of Thing

3 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

[1]. https://xueqiu.com/2596166299/126924156

[2]. “Energy aware performance evaluation of wsns,” Ph.D. dissertation, Middlesex University, December 2014. http://eprints.mdx.ac.uk/17460/

[3]. New LoRaWAN specs allow FOTA upgrades https://enterpriseiotinsights.com/20181025/channels/news/lorawan-specs-allow-fota-upgrades

[4]. A Firmware Update Architecture for Internet of Things Devices https://tools.ietf.org/id/draft-moran-suit-architecture-00.html#rfc.section.3.2

[5]. How to Approach OTA Updates for IoT, https://dzone.com/articles/how-to-approach-ota-updates-for-iot

[6]. OneM2M white paper – The Interoperability Enabler for the Entire M2M and IoT ecosystem. January 2015.

[7]. GTI OPP White Paper_IoT Service Layer Architecure. Jun 2018

[8]. OMA-TS-LightweightM2M-v1_0_1-20170704-A. Jul 2017

4 Challenges and Chances of FOTA for NB-IoT

IoT introduces a vision of a future internet where users, computing systems, and everyday objects possessing, sensing and actuating capabilities cooperate with unprecedented convenience and economic benefits. As with the current internet architecture, IP-based communication protocols will play a key role in enabling the ubiquitous connectivity of devices in the context of IoT applications [2].

Such communication technologies are being developed in line with the constraints of the sensing platforms likely to be employed by IoT applications, forming a communication stack able to provide the required power-efficiency, reliability, and Internet connectivity.

OTA (Over-The-Air) Technology is a technology for remote upgrade of terminal device side module firmware or application software through the air interface of a mobile cellular network (2\3\4G or NB). As IoT devices has some limitation about the power consumption、memory and network quality etc., FOTA (Firmware Update Over-The-Air) capabilities can help IoT devices repair and update the system remotely without those limitation.

In an era when there is no OTA upgrade, terminal devices face the ultimate test: Upgrade or do not upgrade, which is a problem. No upgrade is very hurt, occasional failure exists, system performance is degraded, new features cannot be updated; At the same time, upgrade is complex, widely distributed, large quantity, time consuming, large base, artificial expensive, manual upgrade inefficiency, high technical threshold.

With the large-scale deployment of the Internet of things devices, FOTA now is an important aspect especially for the NB-IoT devices. Most of the NB-IoT devices has very limit resource to run the services, need low power consumption, narrow bandwidth, and most of them run without manual's on-site support and may have no I/O operation for the devices. When we add a new feature or try to fix a bug exist, we may need to find the device and try to flash the new image with hardware connection. Sometime it is impossible task.

FOTA is becoming the irreplaceable part of the IoT system. Which can help devices to upgrade its image over the air and without any on-site operation. With the help of FOTA, the developers can import new technical features and product functions to offset the flaw of the product as time goes on. And the chips, modules, applications can ship their product to market quickly and improve their capability and quality easily more conveniently.

Comparing with the high-end telephone and the others upscale intelligent equipment, the update of FOTA for the IOT equipment need to consider more aspects, including the storage space of device, the network bandwidth, upgrade power consumption, and upgrade packet splitting and other issues. The existing solution in over-the-air updates are mostly fragmented and no single solution exists. Also, manufacturers and/or service providers have unique solutions, which are not inter-operable. Most of the existing smart devices on the market use the way that the devices inquire the upgrade patch initiative on the network and pull them to update. While it is not fit for the equipment of IOT completely as it need device to know when and how to get the image, and it need run a service on the device to check if start the whole upgrade process, it is complicate for the resource-constrained NB-IoT device and will need more power consumption. In this practice, we try to use push method for obtaining the image the condition, it reduce the complication on the device and make a low power consumption to meet the NB-IoT scenario. In this section we will introduce the FOTA Characteristics for the NB-IoT and will try to give some suggestions for FOTA design in NB-IoT devices.

4.1 Diverse IoT Device Types

NB-IoT devices used in different scenario. There are various types of devices, and the hardware configuration and software architecture of these devices may be different. These lead to the fragmentation of the end-to-end FOTA solution.

FOTA needs to consider these fragmented requirements, for different hardware and software differences. FOTA design needs to provide a unified simple interface, including for the system operating interface requirements, such as image writing to the Flash interface, network operation interface, and get a message from the image sent from the network side.

At the same time, in order to apply efficient and convenient interaction without affecting the operation of the business, it is necessary to design a number of interfaces that are already on the platform side of the upper application, these unified interfaces help FOTA become a standard solution for different integrators, adapt to different scene needs, reduce the difficulty of adaptation and time economy and other costs.

In addition, in order to easily access and serve different devices on the platform side, for device identification, device operation. This practice is based on the LwM2M resource model, providing a set of optimized FOTA solution and using the resource model.

In this practice, we have designed a number of sets of interfaces as described above, and the functions of each interface will be described in detail later for readers ' reference.

4.2 Low Data Rate and Narrow Bandwidth

The network bandwidth used by NB devices which is limited and the rate is slow. When design the FOTA solution, these characters have to be considered.

For limited bandwidth and rate, FOTA is required a more efficient compression algorithms that provide a more efficient difference algorithm that narrows the image size as much as possible. At the same time, for the NB network, the HTTP and other protocols for the device consumption is too large, and does not apply to the overall upgrade process management. We selected the Internet of Things lightweight protocol CoAP and LwM2M to management the upgrade process and image download, it can better adapt to NB equipment bandwidth limited rate slow use of the scene, and improves FOTA upgrade success rate.

4.3 Ultra-low Power Consumption

NB-IoT imports some power saver models to reduce power consumption, such as EDRX, and PSM. These technology can improve the efficiency of power amplifier by the way of reducing the peak to average power ratio, by the way of reducing the periodic measurement.

Based on these methods, the NB-IoT has been developed and applied successfully. In practical application, the battery life of the NB-IoT are related to specific business models and the coverage of the terminal. NB-IoT is suitable for low-frequency services, the rest of the time are best to be able to sleep or shut down, so as to achieve low power consumption.

For FOTA, the low-power approach is to design simple, lightweight interaction logic that minimizes most of the calculations, including whether the query meets the upgrade threshold, whether the upgrade process needs to be initiated, and how to get the image required for the upgrade in the cloud. This includes the use of excessive resources by the terminal in the FOTA process, which has an impact on the power consumption of the terminal.

Most of the existing market products for FOTA upgrade, are on the terminal to start a long-running service, periodically proactive and OTA platform query and other interactive services, until the query to meet the requirements of the image to be upgraded, and then by the terminal is active with OTA server to establish a connection. And proactively request access to the entire upgrade image from the cloud pull. The impact on terminal and network interaction is relatively large, not very suitable for NB-IoT scenarios. In our practice, upgrade package query, image issue and most of the high consumption operations are all done on the platform to minimize the impact of FOTA on the terminal.

4.4 Robustness

Robustness refers to the ability of a system to maintain its original state under continuous disturbance. As current situation of NB network is not very well, robustness is a factor that must be taken into account for FOTA processes.

The first point is the breakpoint retransmission. For the reason that the users do not know whether the transfer of the package has been completed, the breakpoint detection will detect the package's integrality. If the packages' integrality can't meet requirement, the breakpoint retransmission will start.

And if the transmission is interrupted uncontrollably, the intelligent reduction can delete the useless and half-baked package to save space and restore the file and package. The new FOTA plan can report the real-time status of NB-IoT devices, fine upgrade status tracking and data analysis of device behavior trajectory. And to ensure that the device dynamics can be mastered at any time. When the breakpoint occurred, it can be continued.

The design goal of NB-IoT is enhancing 20dB coverage basing on the GSM function. And on the down link, it mainly increase the maximum repetition of each channels to achieve coverage enhancement. The coverage radius of the NB-IoT is four times as big as GSM/LTE. And the more coverage means more user equipment, which means harder to manage.

The FOTA need to consider the minimal performance access device to sub package. The package should separate properly . And for some devices, the storage space are limited, the package itself should be small enough.

The power-off protection from FOTA is also necessary. As the users are unable to know whether the NB-IoT is still working. With the power-off protection, the users' losses can be minimized. FOTA need check the information of data point. Let the data point be continued transmit, which avoid the equipment interrupt and restart.

4.5 Security

A strategy for end-to-end security is crucial, to ensure the device and the data are protected, hence ensuring confidentiality, integrity and availability to the user. These elements of the triad are considered the three most crucial components of security.

The FOTA of IoT devices must consider upgrade security. Many IoT devices are infrastructure and are unattended devices, if they are attacked, the attack cannot be aware timely and it often affects large batches of users. To ensure that the upgrade security is a very important part of the FOTA.

To ensure security during the upgrade process, FOTA needs to take a variety of means to implement authentication and security link mechanisms, upgrade package signing and verification mechanisms, upgrade package encryption and decryption mechanisms, and, during the upgrade process. FOTA also needs to symmetric encryption algorithm, asymmetric encryption algorithm, client storage data encryption (rc4) and a series of measures to protect core data security, and use the network identity authentication, IP address filtering and other measures to prevent protection from external attacks.

When the new upgrade version is released, the third-party security signature is first used, the public key is encrypted and uploaded to the OTA server through a private encryption algorithm, and the server downloads the upgrade file to the client through a dedicated security link, the client performs the integrity operation on the received upgrade file, the signature verification and the private key is decrypted and enters the upgrade. At the same time, in order to ensure the stability of the server, in the upgrade process, it need to join the server early warning mechanism, regular security reinforcement of the server and firewall devices.

Two-way identity authentication ensures that the uniqueness of the internal level of NB-IoT can be not tampered. It can prevent devices and networks from being maliciously damaged. And the entire process is initiated and controlled by the OTA server.

To prevent the middleman from sending the tampered and forged upgrade package to the device, the terminal needs to add a verification mechanism in the upgrade process.

FOTA services also provide and deploy security servers to protect against unknown types of attacks. It always based on the security server upgrade package reinforcement function. It provide security infrastructure such as key generation and management, digital encryption, digital signatures, and so on.

The FOTA can use the encryption algorithm, signature or multiple encryption to protect each transmission of FOTA is not broken. And each upgrade progress is not be tampered.

5 Cloud to Device Architecture

NB-IoT is featured with wide-area coverage than other proximity IoT technology0 . An IoT cloud platform is capable to managed NB-IoT devices directly without an intermediate gateway. So the two stages FOTA update procedures can be achieved directly between NB-IoT devices and the IoT platform:

I. Firmware Image transmission from IoT platform to device.
II. Device updated with the policy configured directly by the Platform.

Figure 5-1 takes the architecture we adopted in the China Mobile IoT FOTA services as a typical example of such Cloud to Device direct Architecture.

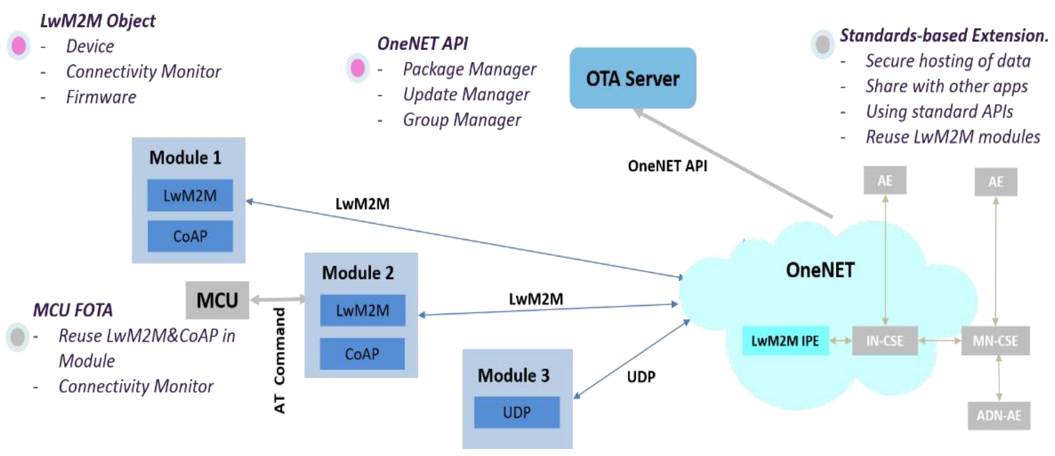

Figure 5-1 Architecture Example of Cloud to Device Directly for M-IoT

There are two typical types of NB-IoT devices described which connect to OneNET over NB-IoT directly. One is type (A) working in host less mode showed in Figure 5-1 Module 1 and module 3, another is type (B) which an additional MCU connects to a NB-IoT module showed as module 2 in figure 5-1.

And there are 3 steps in the end-to-end solution. The first step is preparing, in which we need to prepare the original and target firmware package, making the incremental package online of offline, uploading the package to OTA server, choosing the target group, target devices. The Second step is updating. In this step, server needs to check the devices status to make sure it meet the update requirement such as the version, battery voltage, network quality, available flash space etc. It also manages the whole upgrade process including start、download、Integrity check， retry strategy etc. The last step is after the upgrade. As most of the device will restart after upgrade, server will know the upgrade status only after device reconnect. So device need to sync its firmware status to server after reboot and finish the whole process.

Firmware images for the NB-IoT module or the module + MCU are stored and managed on IoT open platform or on the third party OTA server which will use the FOTA services provided by IoT platform. NB-IoT devices can pull the image from the IoT platform directly or the firmware can be pushed to the NB-IoT directly.

With the OneNET implementation, images are managed on OneNET and the method of “PUSH” is used to get the firmware image from OneNET to M-IoT devices. Third part OTA server can use the OneNET FOTA services via OneNET APIs as showed in figure 5-2, and such APIs is also looking forward to be standardized solution in future.

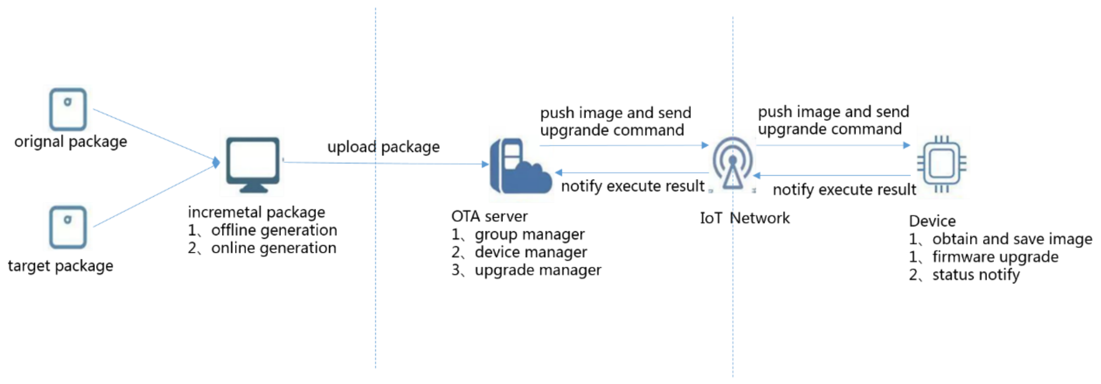

Figure 5-2 FOTA Capability to 3rd Party OTA Server

Usually, the IoT platform would support different protocols to complete the FOTA, some are standardized solution, and some are proprietary depending on the IoT devices capability. While using standardized solution will be the trend for boosting the IoT0 **错误!未找到引用源。**. With OneNET implementation LwM2M & CoAP were recommended as a well standardized solution to transmit the image.

So far, unicast is supported by this OneNET implementation. While considering to minimize the cost of a group devices firmware update, the architecture should be broadcast friendly **错误!未找到引用源。**, this would require the underlayer network to support broadcast/multicast, and might need specific edge computing functions.

6 LwM2M Based FOTA Mechanism

LwM2M is a device management protocol that allows the remote manipulation of constrained devices in the Internet of Things, IoT. LwM2M uses CoAP, Constrained Application Protocol as a transport mechanism.

The OMA Lightweight M2M (LWM2M) is targeted in particular at constrained devices, e.g. devices with low-power microcontrollers and small amounts of Flash and RAM over networks requiring efficient bandwidth usage. At the same time, LWM2M can also be utilized with more powerful embedded devices that benefit from efficient communication. LWM2M provides a light and compact secure communication interface along with an efficient data model, which together enables device management and service enablement for M2M devices.

The upgrade process is managed using Object defined by LwM2M. There will be 3 objects used in LwM2M, Device object, Connectivity object, Firmware Object. In our practice, we expand LwM2M protocol for FOTA, such as add some new resource items used in FOTA, and we also specific some new attributes for the object, we used this definition and expand them to meet our requirement.

For firmware package obtain, we take"PUSH" method, and we defined some APIs to get necessary information and operation from the device system and network layer. To make sure the application deployed in MCU work well during upgrade process, we add some AT command to notify MCU the current upgrade process.

6.1 Firmware Image Download Process

It supports two types of image download method In LwM2M, one is "PUSH" by server and another is "PULL" from device, in this practice, we take "PUSH" method. And the"PUSH"

```
Server                                                      Client
  |                                                            |
  | CON [MID=1], POST, /5/0/0, 1:0/1/128       ------>         |
  |                                                            |
  | <------   ACK [MID=1], 2.31 Continue, 1:0/1/128            |
  |                                                            |
  | CON [MID=2], POST, /5/0/0, 1:1/1/128       ------>         |
  |                                                            |
  | <------   ACK [MID=2], 2.31 Continue, 1:1/1/128            |
  |                                                            |
  |              ... 637 exchanges ...                         |
  |                                                            |
  | CON [MID=640], POST, /5/0/0, 1:639/0/128   ------>         |
  |                                                            |
  | <------   ACK [MID=640], 2.04 Changed, 1:639/0/128         |
  |                                                            |
```

process show in below figure 6-1:

Figure 6-1 image Push process

OTA server writes upgrade packages to / 5/0/0 through “Write” request of LwM2M, it can be explained that in package resource of the 0 instance of Firmware Update object. As upgrade package often exceeds the limit of one message, it needs to be divided into multiple messages. For example, the size of upgrade packages is 80KB, and the payload size is 128 bytes. The entire upgrade package will be distributed to 640 messages. It will be write to device one by one. After the device gets the entire upgradable package, the platform will check the current status of the device about the battery level and network quality. If the upgrade condition is satisfied, the upgrade request will be issued, and the device will manage the upgrade process according to the specific instructions.

Compared to “PULL”, “PUSH” from the server is more appropriate for M-IoT use scenarios.

First, “PUSH” simplifies the calculation requirements of the device side, less the device resource consumption and power consumption. “PUSH” from the server puts complex logic on the OTA server, and for resource-constrained M-IoT devices, the simple implementation is an important factor in the success of the final deployment.

Second, “PUSH” is more appropriate for the current trend of M-IoT. With “PUSH”, the entire upgrade process is managed by server, including image downloads. Server can optimize the process such as subsequent dynamic loads of packets based on a wider range of information.

Third, “PUSH” can handle abnormal situation processing more effectively. Compared to the devices, server is always online. Dealing with these anomalies in the server can better solve such as failure retransmission, breakpoint continuation and other functions, and it can improve the success rate of the upgrade.

6.2 Continuous Transmission from Breakpoint

For NB-IoT network, the bandwidth limitation of the network and the capacity of the terminal devices are limited. The network side will split the upgraded image into several message packets. Because of the poor coverage of the Internet of Things, the upgraded packets will probably be interrupted in the process of pushing the upgraded packets, especially in the case of mobile devices. If the data packet transmission is interrupted, the content of the data packet that has been received before cannot accurately get the data point of the current interruption. When the network connection restores and the process starts again, only the stored information can be discarded and the whole upgrade image transmission can be restarted, which results in a lot of waste of resources. At the same time, for the low coverage level scenario of the Internet of Things, the coverage cannot be maintained throughout the package transmission process, it needs to start again after each interruption, resulting in the ultimate failure to complete the upgrade process.

We designed continuous transmission of breakpoint solution. We should provide the ability of breakpoint continuous transmission when downloading interrupts during the whole upgrade package process. Whether in the server active “PUSH” scenario or in the device active “PULL” scenario, the device is required to record the split data point of the last message that has been successfully received each time the data packet is retained. When a new data packet Push or Pull operation begins, the information of the data point should be checked, and the upgrade package after the data point can be obtained according to the information of the data point, so as to achieve the ability of breakpoint continuation. When the terminal receives the CoAP sub package, it should update/5/0/26500 (representing the number of bytes that the current device has just stored), check the status on the platform side and read/5/0/26500 to get the storage status on the device side when retransmitting, so as to find the breakpoint of the upgrade package and transfer the remaining bytes of the firmware package from the next byte of the transmitted byte.

6.3 State Machine

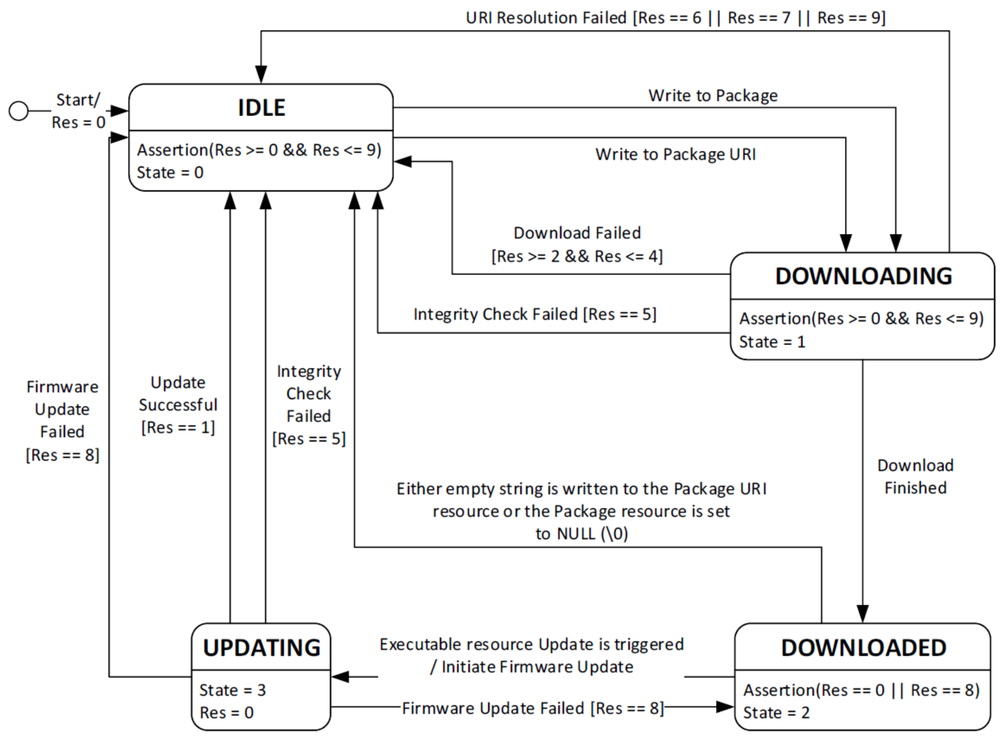
URI Resolution Failed [Res == 6 || Res == 7 || Res == 9]
Start/ Res = 0
IDLE
Assertion(Res >= 0 && Res <= 9)
State = 0
Write to Package
Write to Package URI
Download Failed [Res >= 2 && Res <= 4]
Integrity Check Failed [Res == 5]
DOWNLOADING
Assertion(Res >= 0 && Res <= 9)
State = 1
Firmware Update Failed [Res == 8]
Update Successful [Res == 1]
Integrity Check Failed [Res == 5]
Either empty string is written to the Package URI resource or the Package resource is set to NULL (\0)
Download Finished
UPDATING
State = 3
Res = 0
Executable resource Update is triggered / Initiate Firmware Update
Firmware Update Failed [Res == 8]
DOWNLOADED
Assertion(Res == 0 || Res == 8)
State = 2

Figure 6-2 State machine

We use standard state machine defined in LwM2M, described as in state diagram. The state diagram consists of states, drawn as rounded rectangles, and transitions, drawn as arrows connecting the states. A trigger is an event that may cause a transition, guard is a condition and behavior is an activity that executes while the transition takes place. The states additionally contain a compartment that includes assertions and variable assignments. For example, the assertion in the IDLE state indicates the value of the "Update Result" resource (abbreviated as "Res") must be between 0 and 9. The State resource is set to zero (0) when the program is in this IDLE state.

Errors during the Firmware Update process MUST be reported only by the "Update Result" resource. For instance, when the LwM2M Server performs a Write operation on resource "Package URI", the LwM2M Client returns a Success or Failure for the Write operation. Then if the URI is invalid, it changes its "Update Result" resource value to 7.

7 Effective Incremental Algorithm

There are two types of upgrade packages, one is the entire package, which is compiled directly from the system code, and it will be flash directly to replace the original program on the device. This package is large, there are hundreds of megabytes, but not depend on the current phone software version. Another upgrade is the Incremental package, which is a difference between two software versions, we said it was the previous version's patch. This upgrade package depends on the original version of the device. For IoT devices, due to bandwidth and the limitations of the device's own resources, it is more economical and feasible to take differential packet upgrades.

Incremental packages are most generated by specialized manufacturers now, including the differential packet restore algorithm that needs to be used at the time of the upgrade, during which the upgrade script detects fingerprint and ensures that the upgrade package is applied correctly.

For FOTA, efficient differential reduction algorithm is an important aspect of upgrade efficiency. At present, the differential algorithms commonly used in the market including xdelta(http://xdelta.org), Courgette(http://www.chromium.org) and Bsdiff(http://www.daemonology.net/bsdiff/). With this algorithm, upgrade image can be less than 100KB as the entire image maybe 1MB. It can bring the advantages of reducing the time of empty transmission and reducing the consumption of terminal resources. We need to use an efficient algorithm to generate the incremental upgrade image.

8 Specific Data Model and Interface Design

To support the FOTA, we need define a date model between the UE and OTA server in the update process. In this practice, we use LwM2M's object/resource date model and add some new system API for the developer.

8.1 Object Design and Selection

There are three standard objects in FOTA process, named Device, Connectivity Monitoring, and Firmware Update.

The resources used listed below:

Device Object provides a range of device related information which can be queried by the LwM2M Server, and a device reboot and factory reset function.

Name	Object ID	Instances	Mandatory	Range of enumeration
Device	3	Single	Mandatory	urn:oma:lwm2m:oma:3

Id	Name	Operation	Type	Range	Units	Description
3	Firmware Version	R	String			Current firmware version of the Device. The Firmware Management function could rely on this resource.
9	Battery Level	R	Integer	0-100	%	Contains the current battery level as a percentage (with a range from 0 to 100). This value is only valid for the Device Internal Battery if present.
10	Memory Free	R	Integer		KB	Estimated current available amount of storage space which can store data and software in the LwM2M Device.

This LwM2M Object enables monitoring of parameters related to network connectivity. In this general connectivity object, the resources are limited to the most general cases common to most network bearers. It is recommended to read the description, which refers to relevant standard development organizations (e.g., 3GPP, IEEE). The goal of the Connectivity Monitoring Object is to carry information reflecting the more up to date values of the current connection for monitoring purposes. Resources such as Link Quality, Radio Signal Strength, and Cell ID are retrieved during connected mode at least for cellular networks.

Name	Object ID	Instances	Mandatory	Range
Connectivity Monitoring	4	Single	Mandatory	urn:oma:lwm2m:oma:4

Id	Name	Operation	Type	Units	Description
2	Radio Signal Strength	R	Integer	dBm	This node contains the average value of the received signal strength indication used in the current network bearer. For NB-IoT network bearers the signal strength parameters indicated NRSRP
8	Cell ID	R	Integer		Serving Cell ID in case Network Bearer Resource is a Cellular Network

Firmware Object enables management of firmware which is to be updated. This Object includes installing firmware package, updating firmware, and performing actions after updating firmware. The firmware update MAY require to reboot the device; it will depend on a number of factors, such as the operating system architecture and the extent of the updated software.

Using the object and resource structure defined in this section experiencing communication security protection using DTLS, the envisioned functionality with LwM2M version 1.0 is to allow a LwM2M Client to connect to any LwM2M version 1.0 compliant Server to obtain a firmware image. There are, however, other design decisions that need to be taken into account to allow a manufacturer of a device to securely install firmware on a device.

Examples for such design decisions are how to manage the firmware update repository at the server side (which may include user interface considerations), the techniques to provide additional application layer security protection of the firmware image, how many versions of firmware image to store on the device, and how to execute the firmware update process considering the hardware specific details of a given IoT hardware product. These aspects are considered to be outside the scope of the LwM2M version 1.0 specification.

A LwM2M Server may also instruct a LwM2M Client to fetch a firmware image from a dedicated server (instead of pushing firmware image to the LwM2M Client). The Package URI resource is contained in the Firmware object and can be used for this purpose.

A LwM2M Client MUST support block-wise transfer if it implements the Firmware Update object.

A LwM2M Server MUST support block-wise transfer. Other protocols, such as HTTP/HTTPs, MAY also be used for downloading firmware updates (via the Package URI resource). For constrained devices it is, however, RECOMMENDED to use CoAP for firmware downloads to avoid the need for additional protocol implementations.

Once a new firmware image is successfully installed in the Device, if the Resource "Firmware Version" ID:3 is available in the Instance of the Device Object (ID:3) the LwM2M Client MUST update such a Resource accordingly to the new firmware image version.

Note : PkgName (ID:6) and PkgVersion (ID:7) Resources contain optional information handled by the Firmware Update functionality for its own purpose and have no direct correlation with the "Firmware Version" ID:3 of Device Object ID:3.

Name	Object ID	Instances	Mandatory	Range of enumeration
Firmware	5	Single	Mandatory	urn:oma:lwm2m:oma:5

Id	Name	Operation	Type	Range	Description
0	Package	W	Opaque		Firmware package
1	Package URI	RW	String	0-255 bytes	URI from where the device can download the firmware package by an alternative mechanism. As soon the device has received the Package URI it performs the download at the next practical opportunity. The URI format is defined in RFC 3986.

2	Update	E			Updates firmware by using the firmware package stored in Package, or, by using the firmware downloaded from the Package URI. This Resource is only executable when the value of the State Resource is Downloaded.
3	State	R	Integer	0-3	Indicates current state with respect to this firmware update. This value is set by the LwM2M Client. 0: Idle 1: Downloading 2: Downloaded 3: Updating
5	Update Result	R	Integer	0-9	Contains the result of downloading or updating the firmware 0: Initial value. Once the updating process is initiated 1: Firmware updated successfully, 2: Not enough flash memory for the new firmware package. 3. Out of RAM during downloading process. 4: Connection lost during downloading process. 5: Integrity check failure for new downloaded package. 6: Unsupported package type. 7: Invalid URI 8: Firmware update failed

					9: Unsupported protocol.
8	Firmware Update Protocol Support	R	Integer	0,1	This resource indicates what protocols the LwM2M Client implements to retrieve firmware images. 0 – CoAP (as defined in RFC 7252) with the additional support for block-wise transfer. CoAP is the default setting. 1 – CoAPS (as defined in RFC 7252) with the additional support for block-wise transfer
9	Firmware Update Delivery Method	R	Integer	0-2	Indicate its support for transferring firmware images to the client either via the Package Resource (=push) or via the Package URI Resource (=pull) mechanism. 0 – Pull only 1 – Push only 2 – Both
26500	Transferred Bytes	R	Integer		Indicate UE saved firmware package Bytes
26501	Switch to Download State	E			Indicate UE switch to Download State

8.2 Interaction with External MCU

In case external MCU intervene the process of update, unsolicited information informs the external MCU update state. Please reference AT document [6] for more detail information.

- FIRMWARE DOWNLOADING: Indicates UE is downloading update package.
- FIRMWARE DOWNLOAD FAILED: Indicates download failed.
- FIRMWARE DOWNLOADED: Indicates download finished.
- FIRMWARE UPDATING: Indicates UE is updating.

- FIRMWARE UPDATE SUCCESS: Indicates update success but not report update state to firmware package server yet.
- FIRMWARE UPDATE FAILED: Indicates update failed.
- FIRMWARE UPDATE OVER: Indicates reported update state to firmware package server.

During FOTA procedure, device shouldn’t operate modem until "FIRMWARE UPDATE OVER" is reported. E.g. AT+NRB, power off are not permitted, otherwise, indescribable error will be made. It is better to report the electricity to IOT platform, so IOT platform can check if there has enough power to do FOTA before trigger FOTA.

8.3 Simplified System API for Developer

We specific necessary APIs required in this practice. It provides capabilities about the update process. Including the firmware version, save the firmware package, validate the package, etc.

Below is the list of those APIs:

uint32_t cissys_getFwVersion(uint8_t** version);

Get the version number of the current firmware. It will notify this value to server and check if it has new firmware version.

uint32_t cissys_getFwBatteryVoltage();

Get the voltage of the equipment. It will notify this value to server and check if it meets the threshold value to start the update process.

uint32_t cissys_getRadioSignalStrength();

Get the strength of the current signal. It will notify this value to server and check if it meets the threshold value to start the update process.

uint32_t cissys_getFwAvailableMemory();

Get the available storage space for the current upgrade package. It will notify this value to server and check if it meets the threshold value to start the update process.

bool cissys_eraseFwFlash ();

Erase the storage of the device, the results are returned by the callback from system and notify to server.

bool cissys_checkFwValidation();

Check the integrity of the upgrade package which has been downloaded.

int32_t cissys_writeFwBytes(uint32_t size, uint8_t* buffer);

Write the firmware package to the specified address.

bool cissys_updateFireware ();

Invoke the system interface to start the upgrade process.

int cissys_getFwSavedBytes();

Get the number of stored package bytes. It will use when re-try the transitions.

9 Exposing FOTA Capabilities via Cloud Platform

Currently, the OneNET platform has opened the NB-IoT device access and operation functions, including reading, writing, executing and obtaining the resource list of the device and FOTA.

The upgrade module of the FOTA provides functions of equipment group management, firmware management, upgrade task management, upgrade statistics, test equipment management and so on for platform access devices. And it realizes the integration and integrated management of equipment group creation, upgrade and maintenance.

9.1 Group Management

The platform takes groups as the unit for operation management: each group adopts grouping modular management, which realizes the creation, editing and deletion of groups, maintenance of the list of devices in the group, and FOTA upgrade operation management of devices in the group within the same module.

Different module manufacturers use different chips, their upgrading capabilities are also different. So when creating a new group, you only need to select the corresponding manufacturer name and module model according to the guidance prompt, and the system will automatically associate the upgrading capability of its corresponding chip.

The vendor name and module model are locked when creating a new group, and cannot be edited.

The device list of the group displays all devices under the group, and enters the device list page through the device list entry of the group module.

All parameters for group is listed below:

Name	Explanation
device ID	unique code for the device
device name	user-defined name for the device
IMEI	international mobile device identification code
manufacturer name	manufacturer name of the device
model of the module	model of the module used in the device
version	firmware version of the device
delete	remove the device from the group

List 9-1 parameters of the group device

9.2 Firmware Management

The group was created before the new firmware was added, it is possible to associate the vendor name and module model of the module within the group synchronously, without manual input or selection.

When performance a differential upgrade process, you need to compare two different versions for an incremental upgrade, and user will create the firmware twice, the first is the current version and next one is the target version.

The firmware management interface mainly displays the firmware version information:

Name	Explanation
current version	current firmware version
target version	target firmware version to upgrade
upgrade package type	differential or whole package
upgrade package size	firmware package size
creation time	time of the package be created and uploaded
operation	edit and delete the upgrade package

List 9-2 parameters of the firmware manager

The “differential” module provides the function of obtaining the incremental package. The uploaded upgrade package can only be used to upgrade the device whose firmware is in the selected as current version.

The "whole package" module provides the function of the uploaded upgrade package can effectively upgrade all devices in the group when the current version is "all historical versions".

The selection data for the current version is derived from the version number created when the firmware was created, because the version numbers of the same model modules are different. For example, in the new firmware version to fill in 2.0, the current version of the inside can choose 2.0 here.

After the upgrade package information was selected and confirmed, the user can go back to the upgrade package management information list interface. In this interface, the upgrade package can be edited and deleted.

9.3 Upgrade Task Management

Upgrade task management manage the progress of each upgrade task.

The specific interpretations of the five task states are as follows:

Not started -- the task has not been executed. At this time, you cannot manually start or cancel the task;

Upgrade -- the task is in progress. At this time, the task cannot be edited or deleted, and the information cannot be exported;

Pause - the task is suspended, at this time cannot edit, delete the task, but can restart the task;

Stop -- the task is cancelled. At this time, the task is invalid and cannot be restarted, cancelled or edited;

Complete -- the task has been completed, that is, all devices have completed the "upgrade package download" and "firmware upgrade" two steps, at this time the task has completed its mission, cannot be launched again to cancel and edit.

When the signal strength quality is lower than the set value, the platform will suspend the task of upgrading the equipment automatically in the area.

After setting all contents of the upgrade task, if the device in the device group performing the upgrade task that does not meet the current version of the selected upgrade package, it would be removed.

When the task is not started, all the task operation instructions in the "upgrade package download" and "firmware upgrade" stages will be automatically started by the system according to the set time. At this time, the upgrade task cannot be manually started.

When the task state is in "upgrade", all the task operation instructions on the platform side are suspended (devices already in the "upgrade package download" stage will continue to complete the download step, but will not enter the upgrade step; The device already in the "firmware upgrade" stage will continue to complete the upgrade step). And then continue to issue the last operation instruction. Devices in the "upgrade package download" stage will continue to complete the download step, but will not enter the upgrade step; Devices that are already in the firmware upgrade phase continue to complete the upgrade steps. When the task state is in "upgrade" and "pause", you can cancel the operation.

Name	Explanation
task name	user-defined name for the upgrade task
number of retries	maximum of retries that allowed for a retry upgrade
retry interval	time between two execution
execution time	start time of an escalation task,”--”: start immediately
deadline	due time of the upgrade task
creation time	creation time of the upgrade task
task status	not started, upgraded, suspended, stopped, completed
success rate	successfully upgraded /total of devices for the task
Operation	add, delete, start, pause, cancel

List 9-3 the parameter of the upgrade task management

Detail will give user the detailed upgrade process information of all devices under the task in detail. User can refer to below list.

Name	explanation

device name	user-defined of the device
IMEI	international mobile device identification number
current retry times	X/XX：total number of current XTH retry/retry strategy
online status	online, offline
download status	not start, in process, completed
upgrade status	not start, in process, completed
State	Successful ,Insufficient space, Memory overflow during downloading, Link broke during downloading, Package integrity check fails, Not supported, Upgrade failure, task expires, download fail

List 9-4 parameter of the upgrade task

9.4 Upgrade Statistics

Upgrade statistics show information about upgrades adopted by a vendor or module.

the noun of the list	explanation
device number	module name currently has an upgrade history
module type	module type with an upgrade record
total of the device	total amount of equipment accumulated
total successful upgrades	total of successful upgrades
touchdown	total of touchdowns
cumulative bytes	cumulative download for each upgrade package
uninstalled download	cumulative downloads - total successful upgrades
total upgrade failures	total upgrade failures for each upgrade package
version details	view information for all upgrade packages

List 9-5 parameter of the upgrade statistics

9.5 Firmware Version Checking

The platform side checks that the terminal version is inconsistent with the target version of the task before starting the upgrade process, and then starts the upgrade process. If the version is consistent, the device is set to have been upgraded successfully. After starting the upgrade process, the platform will continue to check the status and result of the device. If the status or result is not zero, write/5/0/0 under the platform. And if the content is empty, the terminal should reset the state, result and/5/0/26500 to erase the downloaded firmware package.

9.6 Package Sending Retry Strategy

Users can set the number of retries and the minimum interval of retries when creating upgrade tasks on the platform. If the number of failures of the terminal exceeds the maximum number of retries of the task, or the time interval between the trigger time of the FOTA process and the last trigger time is less than the retry time of the task, the process will terminate.

10 Pilot Project

The whole solution was applied with a pilot application to manage the electric bicycles. The number of electric bicycles in this city is huge, which is more than 3 million. The management of those electric bicycles had been a difficult problem. It is difficult to find the specific electric bicycle and identify the owner, which affects the management work efficiency.

In this scenario, we installed NB-IoT modules for the electric bicycle. In this module, we use a few sensors, including gyroscopes, smoke monitoring, sound alarms, etc. The data from the sensors are used to monitor the working status of the electric bicycle, the driving status, location information and other data, all data are send to platform through the NB network.

In this way, real-time information collection, intelligent identification, positioning, tracking, monitoring and management can be realized. It achieved the comprehensive improvement of electric bicycle safety management system. It adopts multi-dimensional data acquisition from terminal, and based on the comprehensive analysis ability of the platform, it assists the traffic police department in on-site law enforcement, and effectively promote the comprehensive management of the electric bicycles.

In the entire project, because of the different versions of the hardware involved, the features on the device will be increased or decreased according to demand. At the same time, in order to release this application quickly, we need to take an iterative approach to repair bug, update feature and so on. The modules installed on the electric bicycle does not have the conditions to disassemble for a hard connection upgrade.

Based on these limitations, we integrated the FOTA solution described in this article into this module, which implements the system interface and network interface required by the solution, completes the overall upgrade process management and packet download, storage, integrity check. It also provide a variety of error status management and recovery. It provided a guarantee for the continuous extension of system functions such as electric bicycle fire warning, warning for electric bicycle entering elevator, special industry bicycle monitoring, and analysis of falling and driving habits and behaviors.

The entire project has achieved good results after deployment. With the FOTA capability, the successful electric bicycle comprehensive management project promotes the NB-IoT industry chain to improve the maturity rapidly. At the same time, it fully meets the application scenarios and management requirements.

11 Moving Forward

In the context of continuous functional optimization and continuous iteration of requirements, the IoT platform supports OTA capability is the technical guarantee for the continuous iteration of the current network equipment. We hope that through FOTA technology to provide the module terminal software remote upgrade capability, to improve after-sales service quality efficiency, to reduce after-sales service costs, and to improve product competitiveness.

In this whitepaper, the lightweight IoT connection protocol LwM2M is used to suitable for the upgrade of chip and module firmware. It establishes an upgrade channel between platform and chip module, and helps resource-constrained devices rapidly deploy their FOTA firmware upgrade capability in cellular IoT. At the same time, it considers the limitations of NB-IoT network, provides fault-tolerant mechanisms such as breakpoint continuous transmission, greatly improves the success rate of upgrading, and provides a better user experience. It can promote the use of IoT, big data and other technologies, and also promote the development of related industries.

Looking forward to the future, the upgrade function of FOTA of all kinds of Internet of Things devices should be the standard of devices. The upgrade of FOTA in Internet of Things faces problems that other intelligent devices have not considered before, and the solutions are very different. At the same time, there are more problems to be solved for the upgrading scenario of Internet of Things equipment. How to consume the least resources and bandwidth, provide more secure and reliable upgrading process for equipment, and how to complete the upgrading of a large number of equipment more efficiently are the focus of our follow-up research. At the same time, the fragmentation and differentiation of the amount of devices in the Internet of Things lead to a difficult that one solution can not cover all scenarios and needs to be adapted.

The current FOTA scheme does not support broadcast mode, for a large number of devices at the same time upgrade, there are many problems such as low upgrade efficiency, long upgrade time and so on, waste network resources, later, with the development of technology, we can consider the use of multicast mode for FOTA transmission. With the development of edge computing technology, we can also add to FOTA's subsequent development options, we can put some network delay long operation on the edge calculator, upgrade package can also be distributed and managed by edge server, reduce the network interaction between devices and platforms, reduce the dependence on network transmission, improve the success rate of upgrade.

In summary, FOTA capability will promote the development of the Internet of things from the channel to the Internet of things application, networks and devices, as well as applications, offer more demand for FOTA, and FOTA needs to consider more scenarios and feature upgrades to meet evolving business needs.