Topic title:

DESIGN OF A SECURE FIRMWARE OVER THE-AIR FOR INTERNET OF THINGS SYSTEMS

Authors: Duc-Hung Le and Van-Nhi Nguyen

CONTENT

1. Topic introduction

Reasons for Choosing the Topic

Currently:

- The number of IoT devices is very large and they are installed in many places.
- The need to update features as well as to fix errors.
- Security is essential to ensure the safety of the update process, storage, and control of the firmware versions in operation.

1. Topic introduction

FOTA

- FOTA - Firmware Over The Air is a widely used technology in embedded systems, particularly in systems like IoT, Automotive, and others.
- Function: Updating firmware for devices through the Internet, LoRa, and other methods.

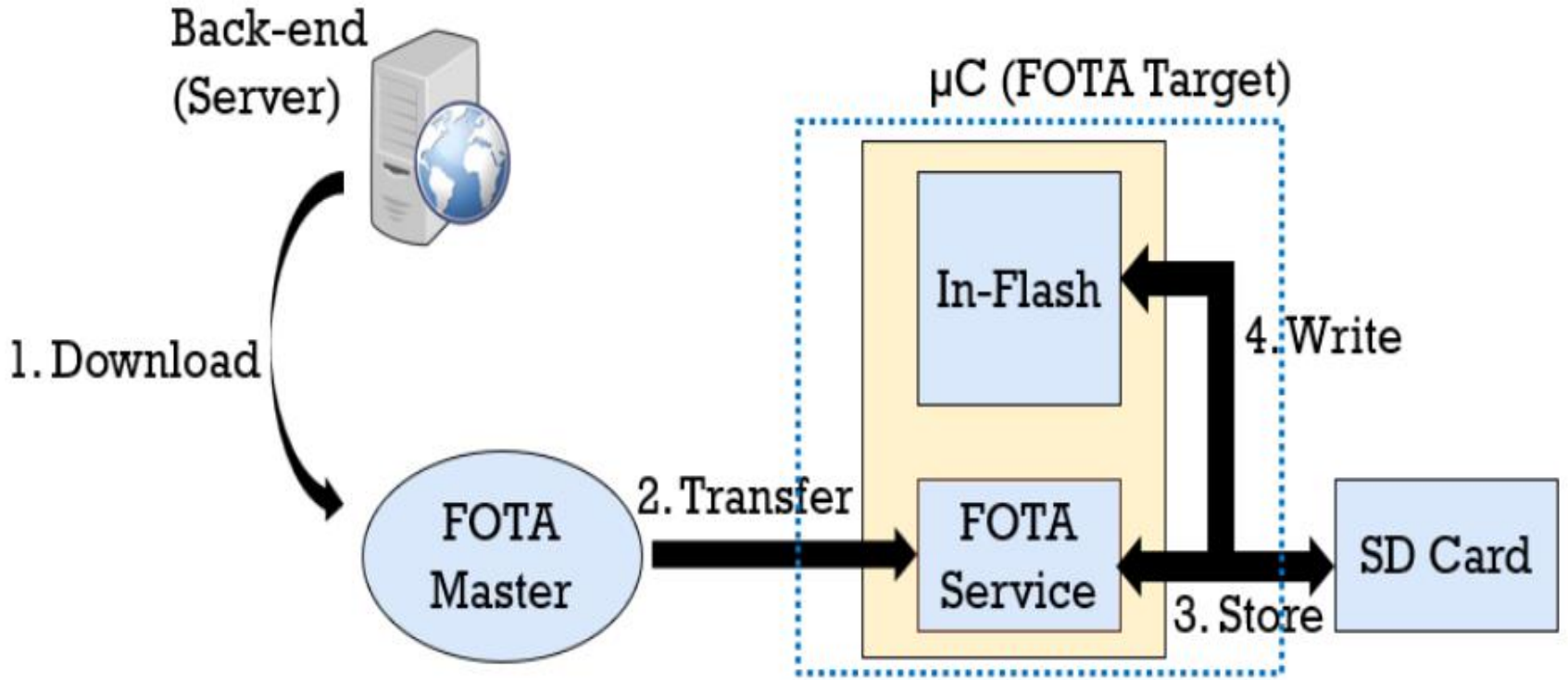
1. Topic introduction

Objective

- Creating a FOTA system with security features.
- Update time, authentication, decryption, and firmware version hashing are within an acceptable range.
- Assessing and providing a development direction for the system (multi-core microcontrollers, FPGA).

2. Theoretical foundation

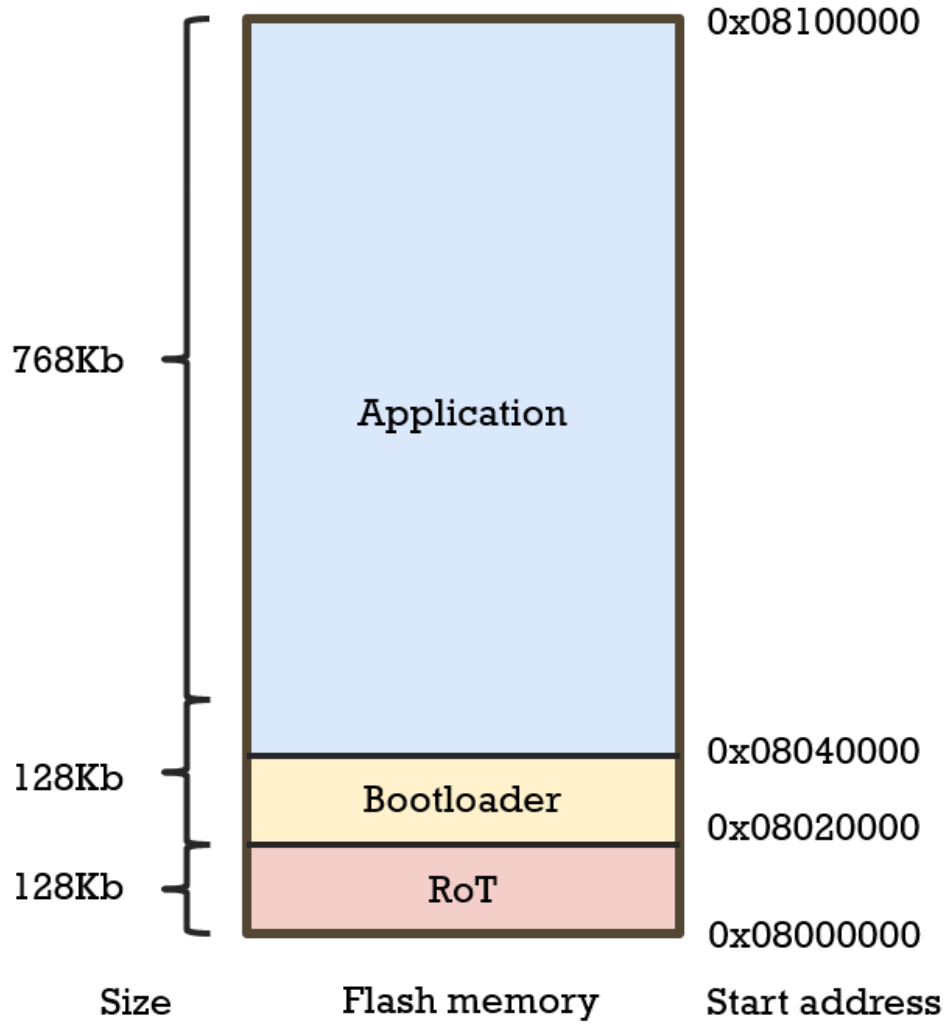
The FOTA system is implemented in the research



2. Theoretical foundation

The Internal Flash memory of a STM32 microcontroller.

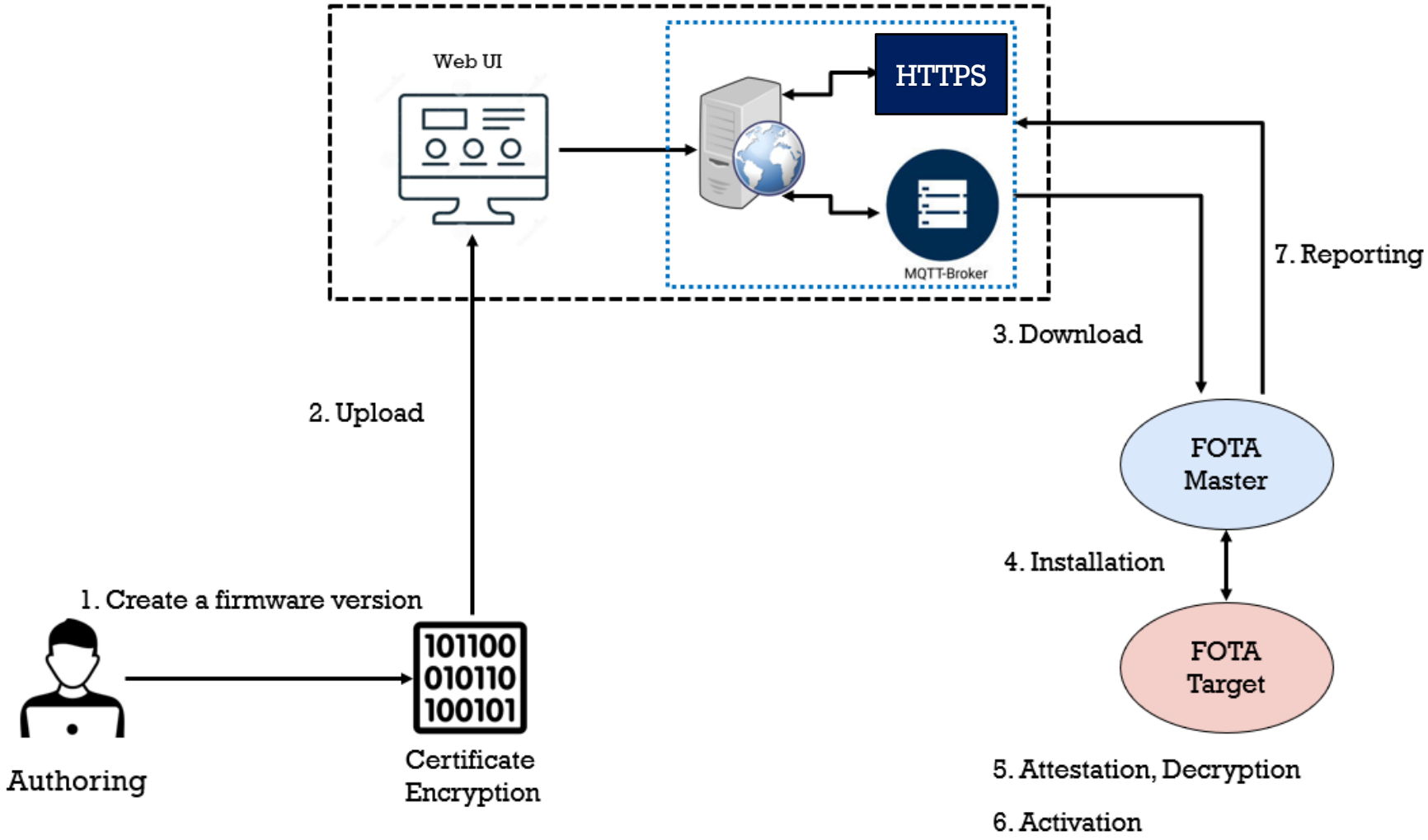
- Size : 1 Mbyte.
- It consists of 12 sectors.
- Program size can be adjusted: start address, size.



(RoT – Root of Trust)

3. Implementation process

Building a FOTA (Firmware Over-The-Air) system



3. Implementation process

Create a firmware version with certification and encryption

Purpose:

- ❑ Certify that this firmware version is genuinely created by us.
- ❑ Ensure its safety during transmission over the network and storage.

3. Implementation process

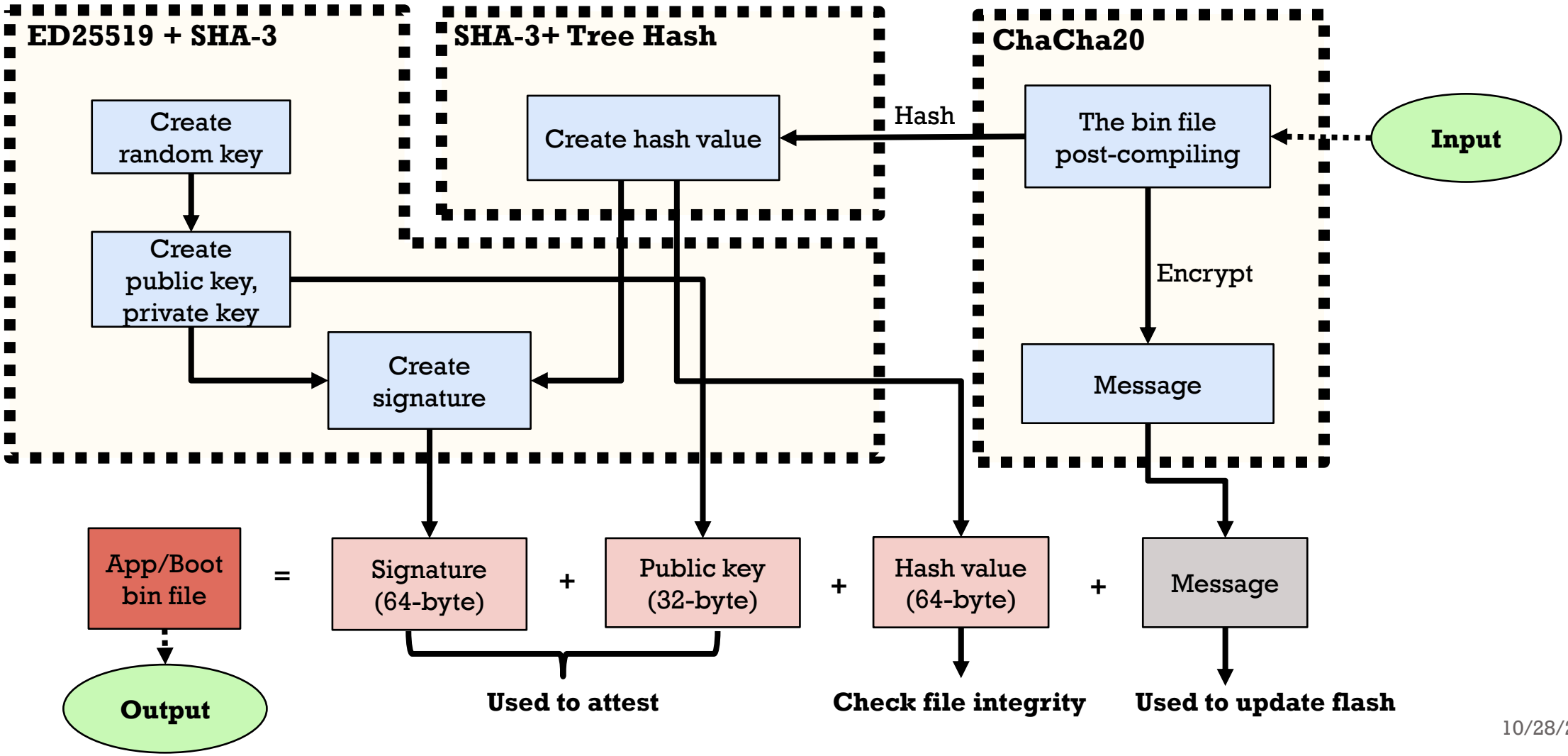
Create a firmware version with certification and encryption

ED25519, SHA3, Tree Hash, ChaCha20 Algorithms

- ED25519: Generating certificates and authenticating firmware versions.
- SHA3: Combined with the ED25519 algorithm, it serves as the means to execute the Tree Hash algorithm.
- Tree Hash: Hashing a large-sized file (firmware version).
- ChaCha20: Encrypts firmware versions and certificates while also performing the decryption of them.

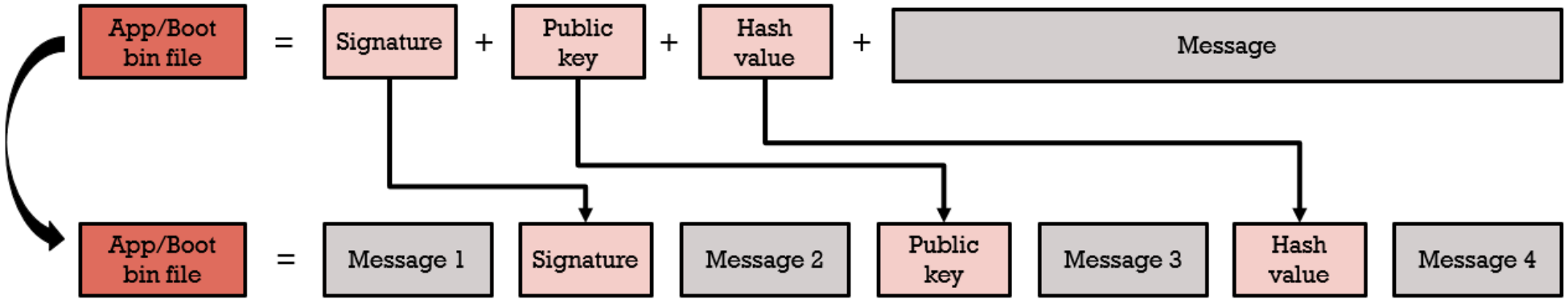
3. Implementation process

Create a firmware version with certification and encryption



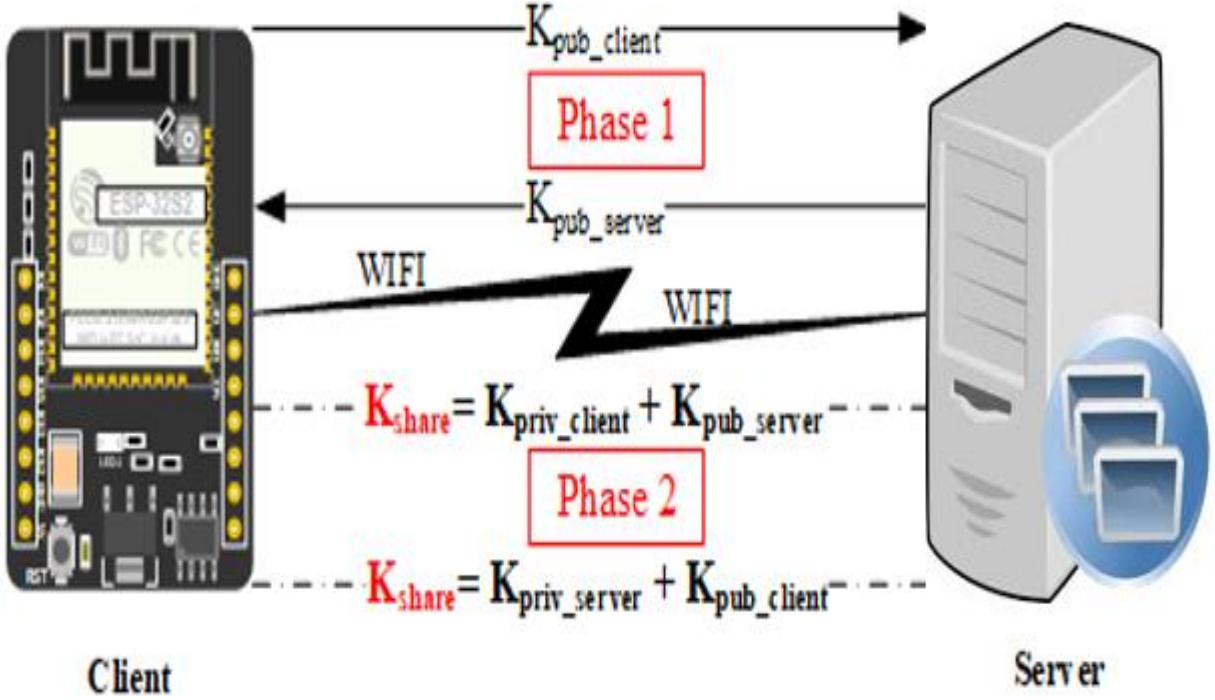
3. Implementation process

Create a firmware version with certification and encryption



3. Implementation process

Deployment Model for Key Exchange Using the ED25519 Algorithm

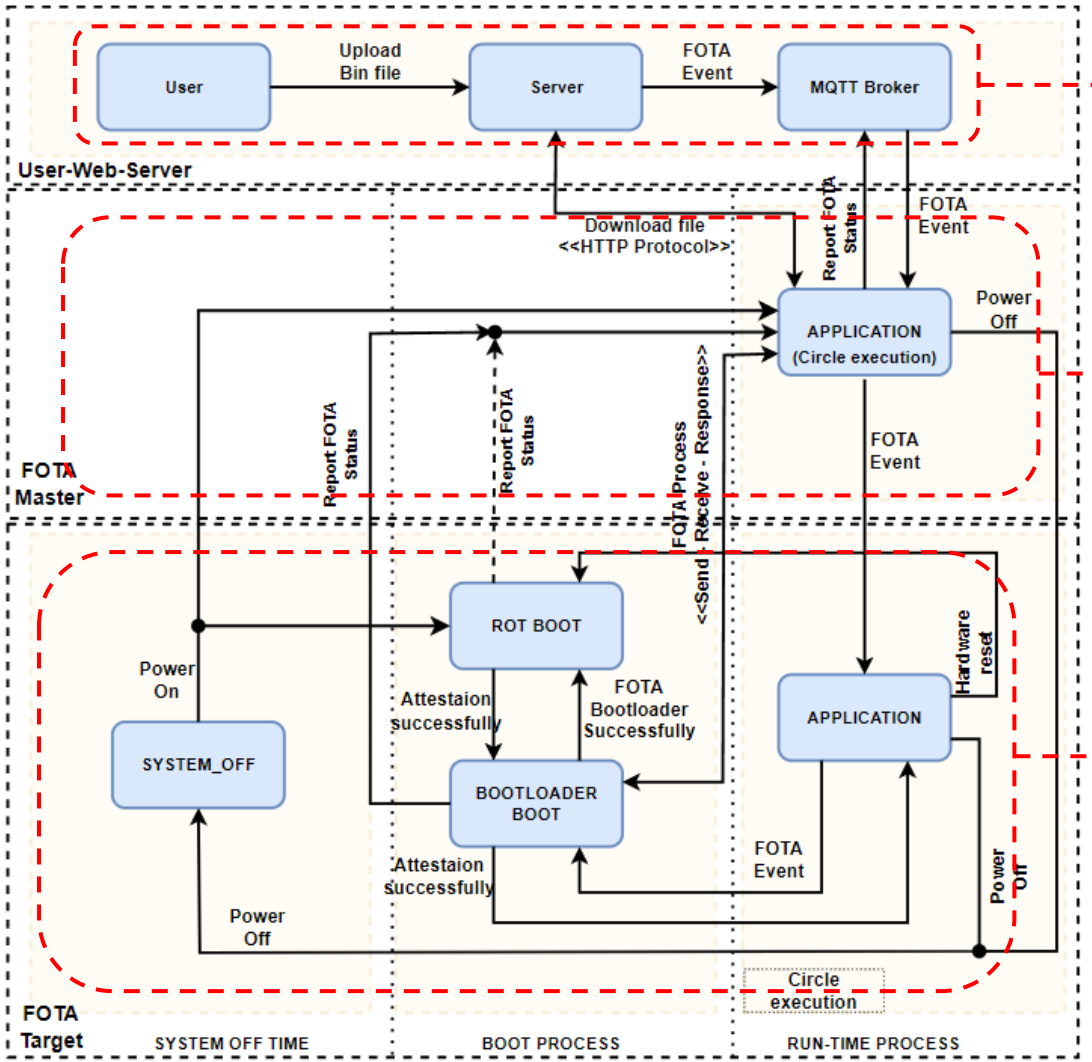


Key Exchange Model with 2-Phase Process:

- Phase 1: The client and server generate and exchange public keys from secret keys using the ED25519 algorithm.
- Phase 2: Create a shared secret key from the received public keys and the secret key based on ED25519.

3. Implementation process

Building a FOTA system



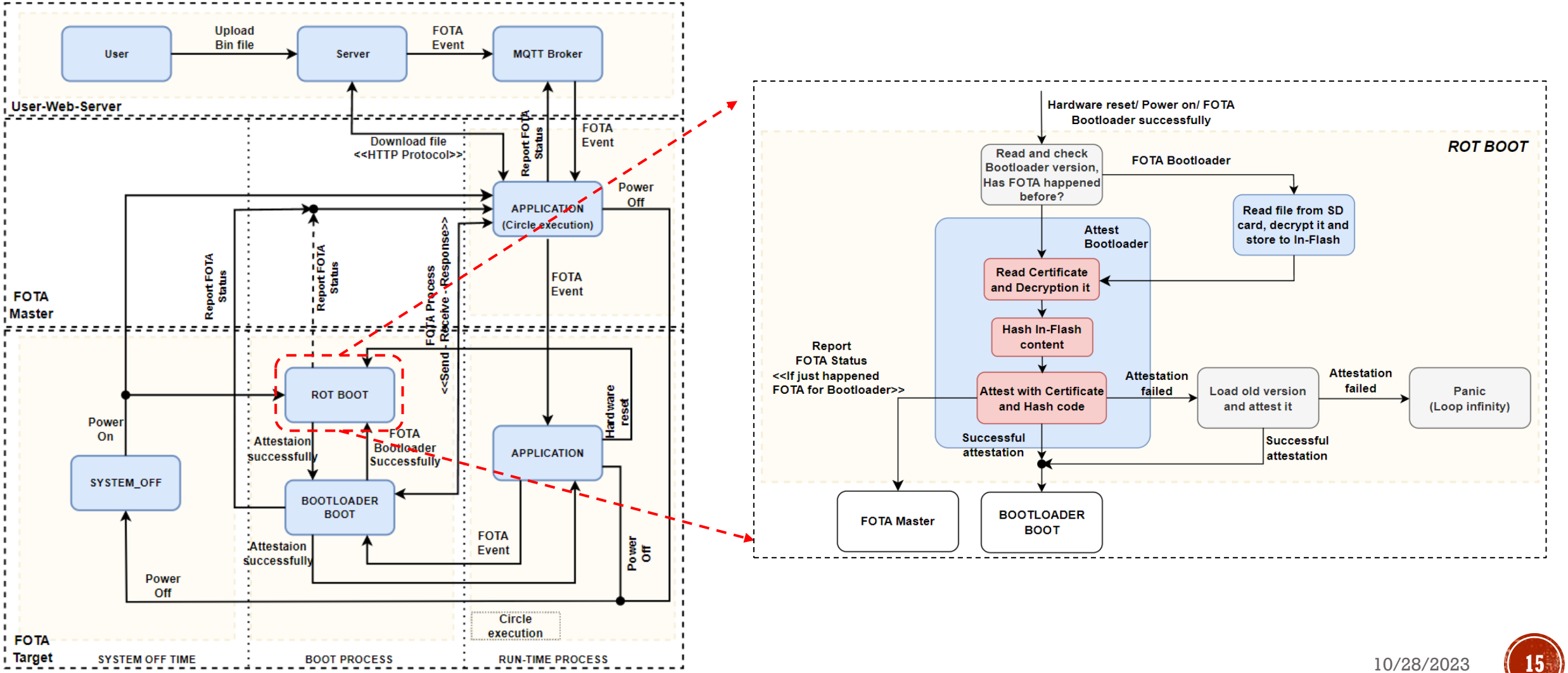
User-Web-Server block

The FOTA Master block (ESP32)

The FOTA Target block (STM32)

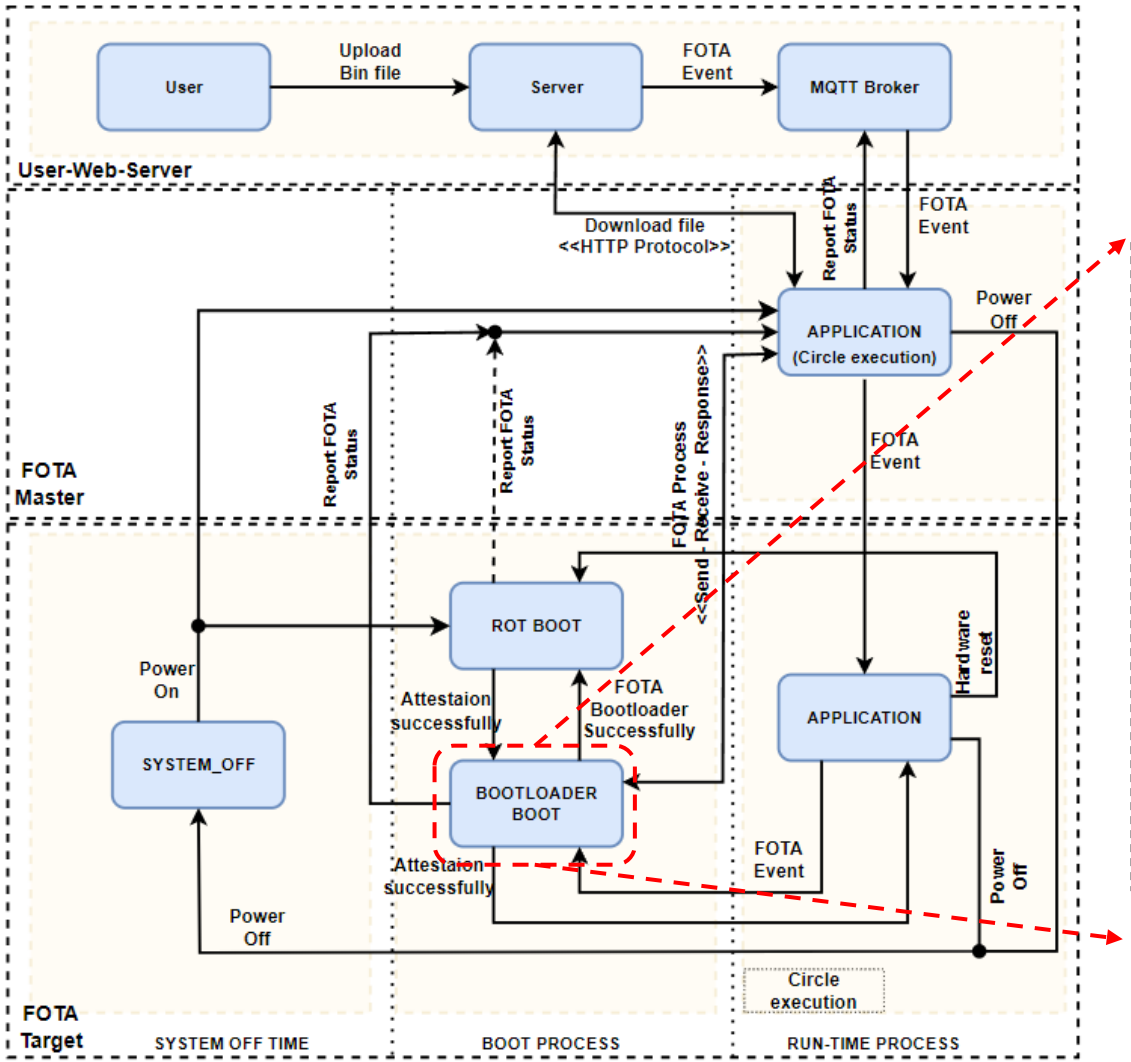
3. Implementation process

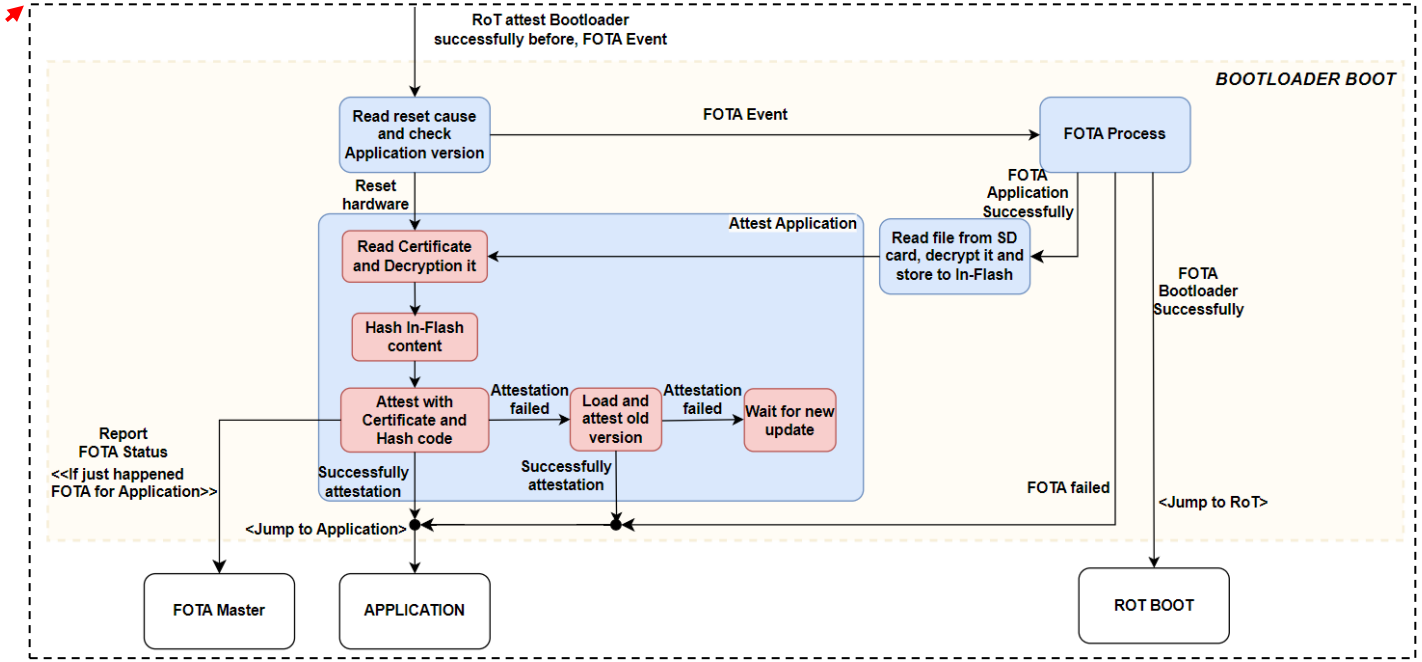
Building a FOTA system - RoT Boot process



3. Implementation process

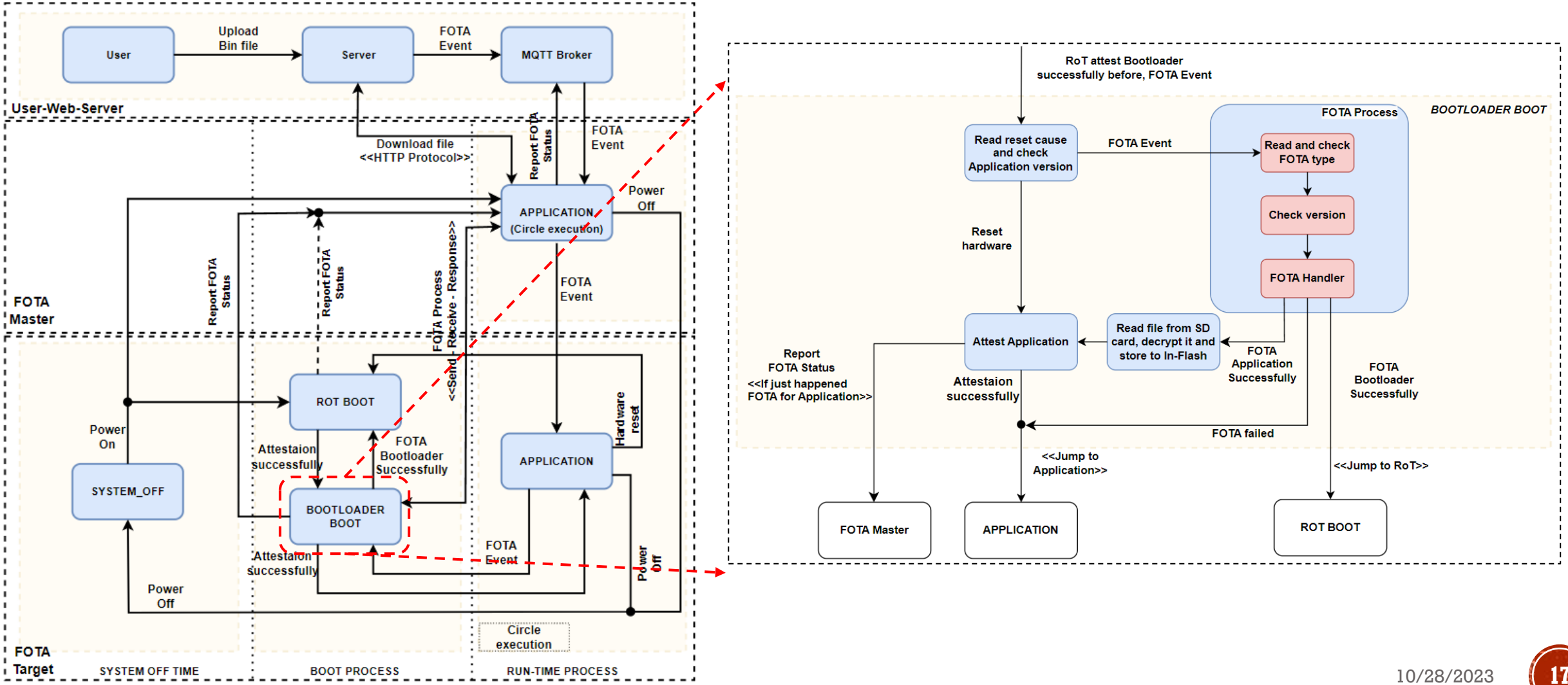
Building a FOTA system - Bootloader Boot process (Application authentication)





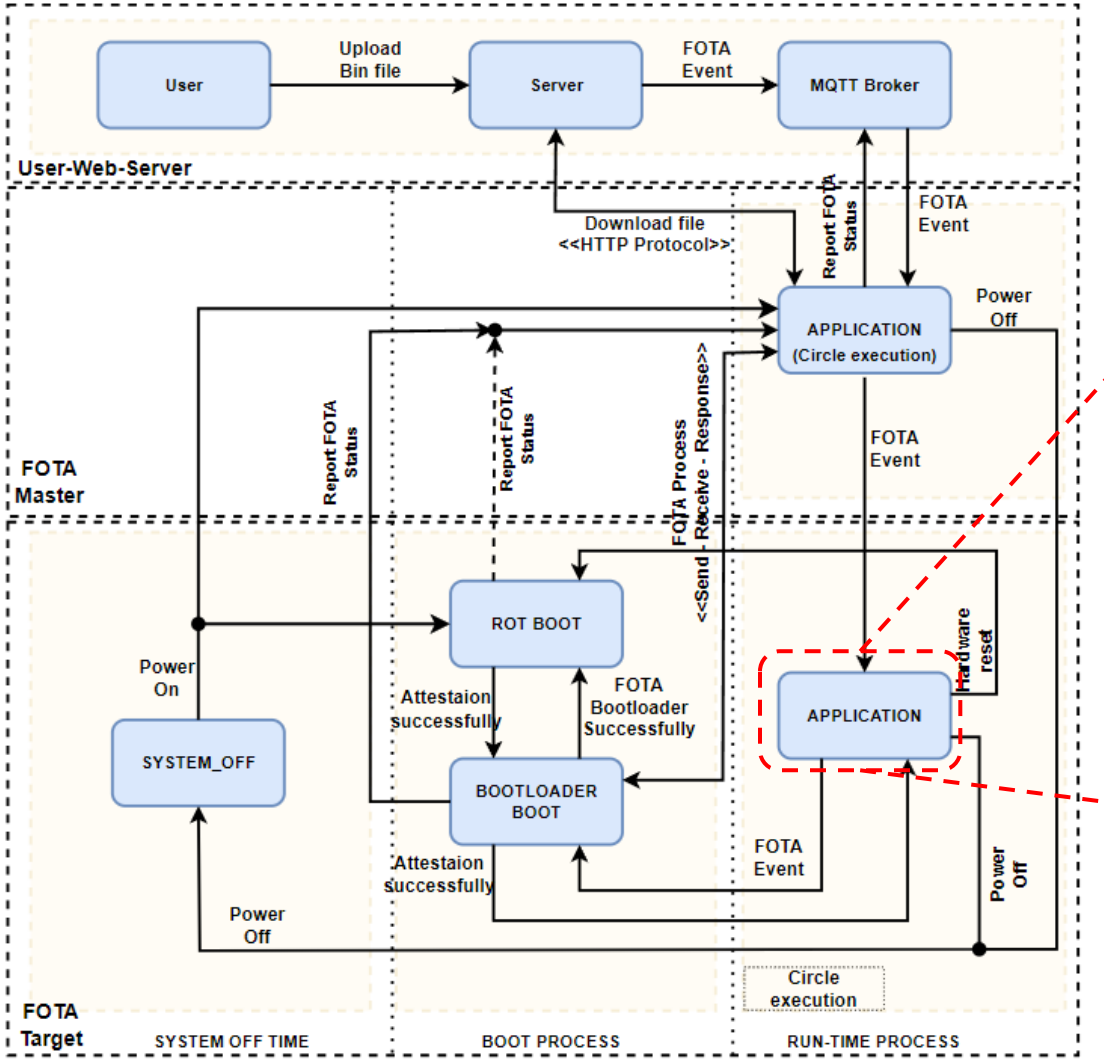
3. Implementation process

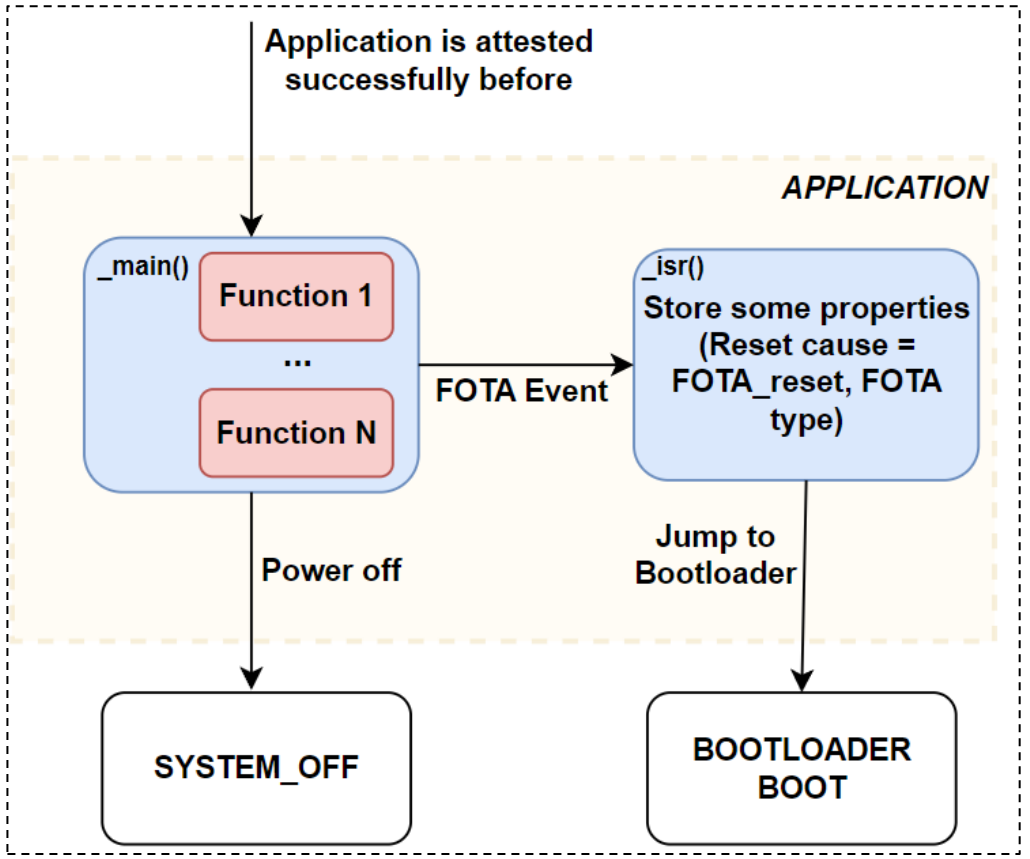
Building a FOTA system - Bootloader Boot process (FOTA)



3. Implementation process

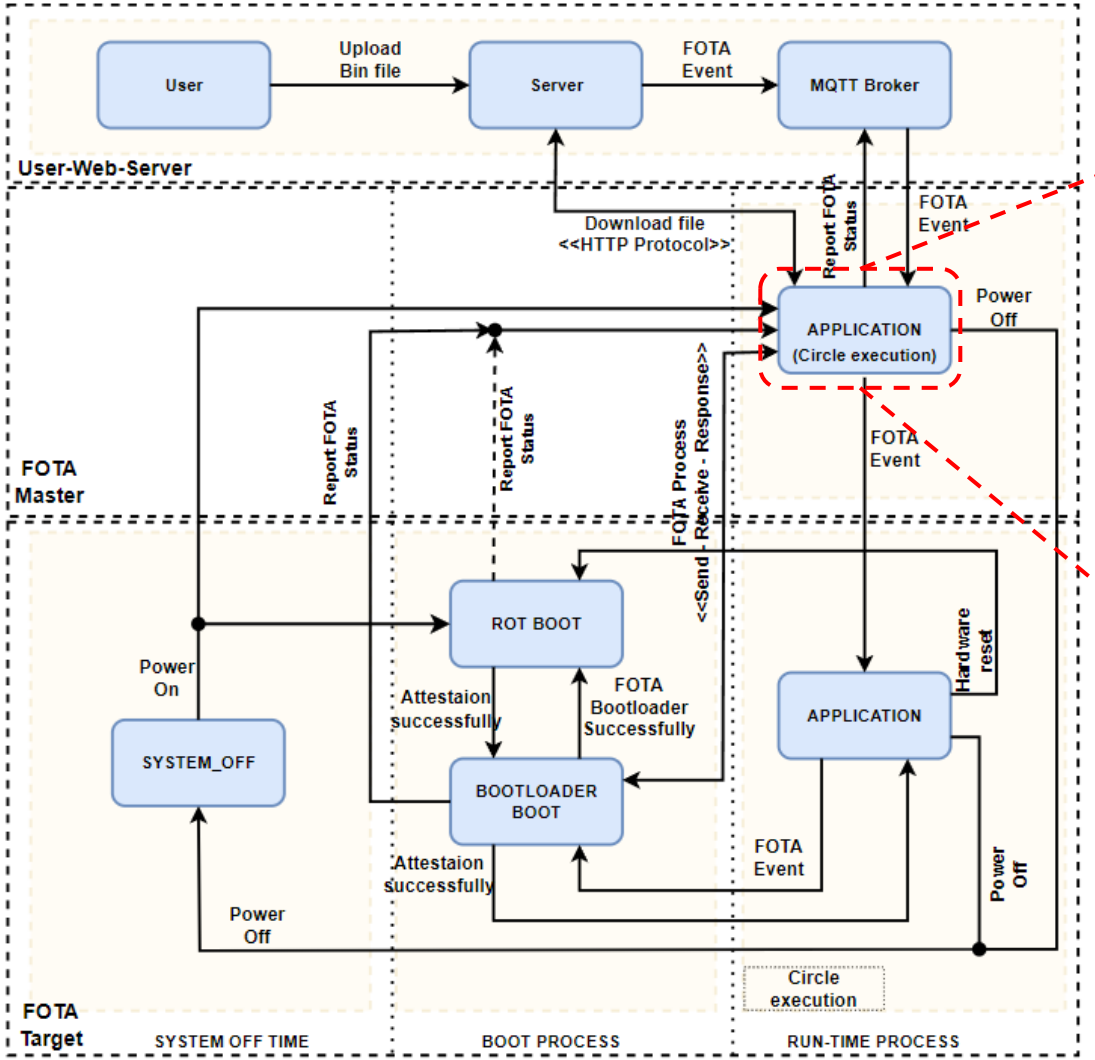
Building a FOTA system - Application Program (STM32)

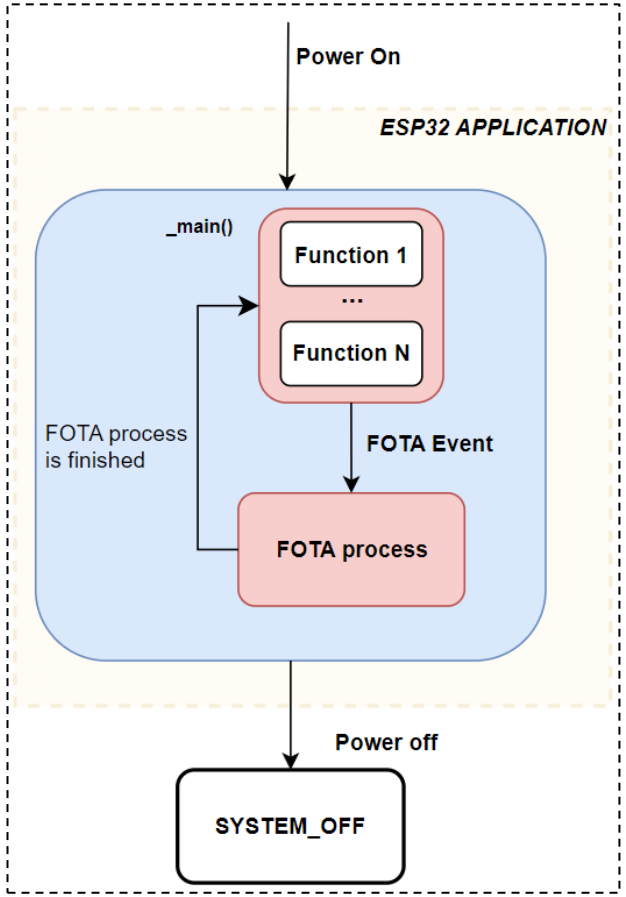




3. Implementation process

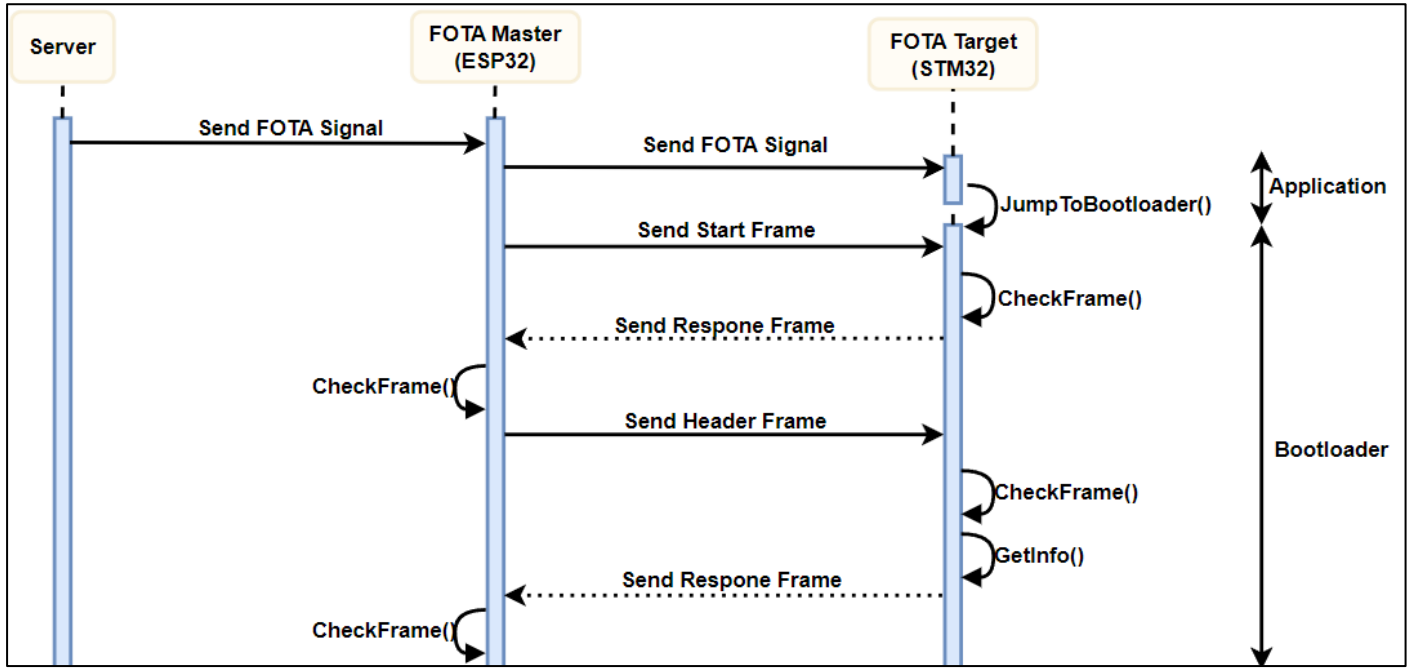
Building a FOTA system - Application Program (ESP32)





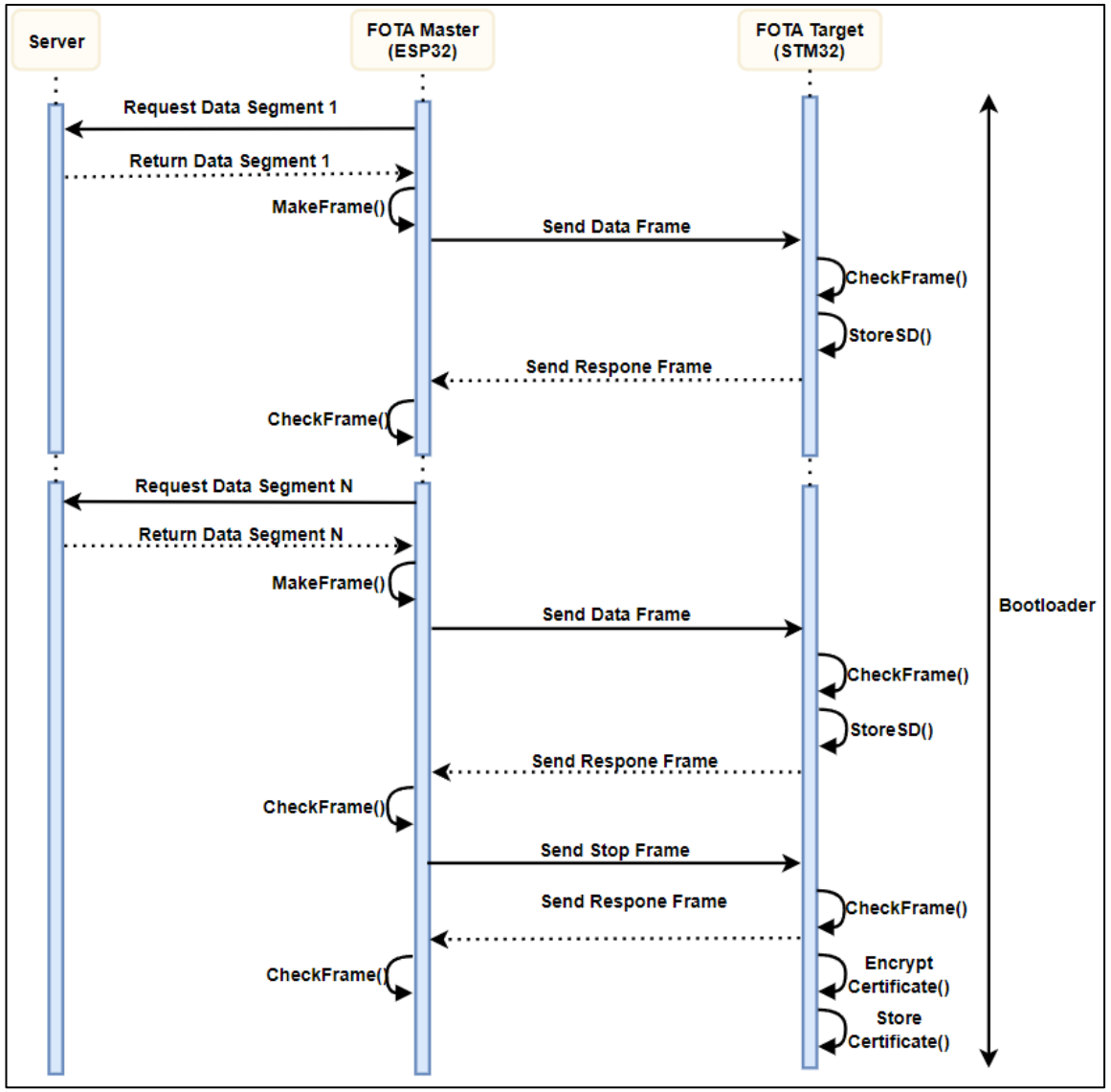
3. Implementation process

Building a FOTA system - FOTA Process (Phase 1)



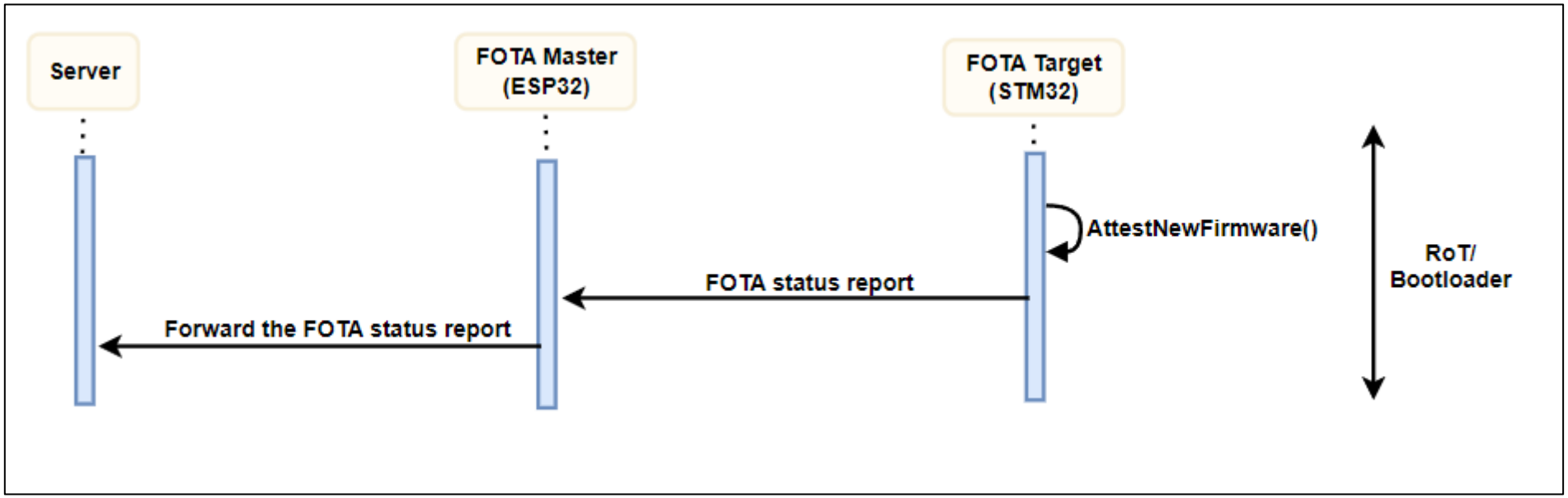
3. Implementation process

Building a FOTA system - FOTA Process (Phase 2)



3. Implementation process

Building a FOTA system - FOTA Process (Phase 3)



4. Implementation results

Create a certified and encrypted firmware version - Bootloader

```
C:\Embedded_System\KLTN\CODE\DEVC\ED25519_SHA3>ED25519_SHA3.exe "C:\Users\Admin\STM32CubeI
DE\RoadToBootloader\BOOTLOADER_ED25519_SHA3_new\Debug\BOOTLOADER_ED25519_SHA3_new.bin" "D:
\BOOTB.bin" BOOTLOADER
Size of the Bootloader file: 127184-byte

Hash value of the file: 47D112014D19EC6239277E0B202CFDF7BB3F38F530BCE3A7EC995C3BFB2CE298
                        6DC4F82BAB5AF3D3D1230FDD35E283AA8BAB5E80BDA74CB02604818E606950B7

Random key:             06E8AFD614C2893ACA39933E568D0ED4D512956FAA95EA5CA2C16A9E338BDCBA

Private key:            F0ED60750F575FDB4D8B0B60EF92FB08A7740C441A74A619845672BE5BFB436E
                        F8920A8541DBAF02FCC2541596BE7882683E33310F028481EF726CF312440A5C

Public key:             0F72D89EA19B36B0539AAF7B70BE2D01E4A120C2FFA92E73F626888FBC812B52

Signature :             3E6B2F7301F03EA81C0C4909575803C2C33F29D968607760C507A895AB6CC13E
                        A22614213CA343054EAC2139FE558CE18499A614C96990600D88C7D223F6920E
```

4. Implementation results

Create a certified and encrypted firmware version - Application

```
C:\Embedded_System\KLTN\CODE\DEVC\ED25519_SHA3>ED25519_SHA3.exe "C:\Users\Admin\STM32CubeI
DE\RoadToBootloader\APPLICATION_F407\Debug\APPLICATION_F407.bin" "D:\APPB.bin" APPLICATION

Size of the Application file: 19716-byte

Hash value of the file: F53BD588E50E4AAF96FD5BB35C5C3216CA962CEFC7E2B2D8B4D5C3A031B43700
                        92B37841C344AF92DB6FFFD49C493E2593F8442382191915O182CA5F8C277669

Random key:             839ADBE2E6B87FFA4BDCF85BB6D417DD53E766A9FBB08CEAB5CE058871412FA6

Private key:            18606A6AA5A40A211D54A2290A98FC54A1A1ECB1AFB35D2F38D8FCC1A9E8784D
                        AB6841FCF9A27BA6C70F8634620A427943227B42CCAF6B8A1BDD4C62D7E6441E

Public key:             06DECBC031B663DE893D25E70B842675E7F8EFB204A383C9FC97EB7991B5DB75

Signature :             416D1929B5EA4A15F6049DFA182C0671482ED026044DA2E12B4A33195EF7EFA8
                        8E255454145B43961C0ED39F20EFB1D6ACE4250AC36A8B9F1F8EA4F0B7D08507
```

4. Implementation results

The boot process

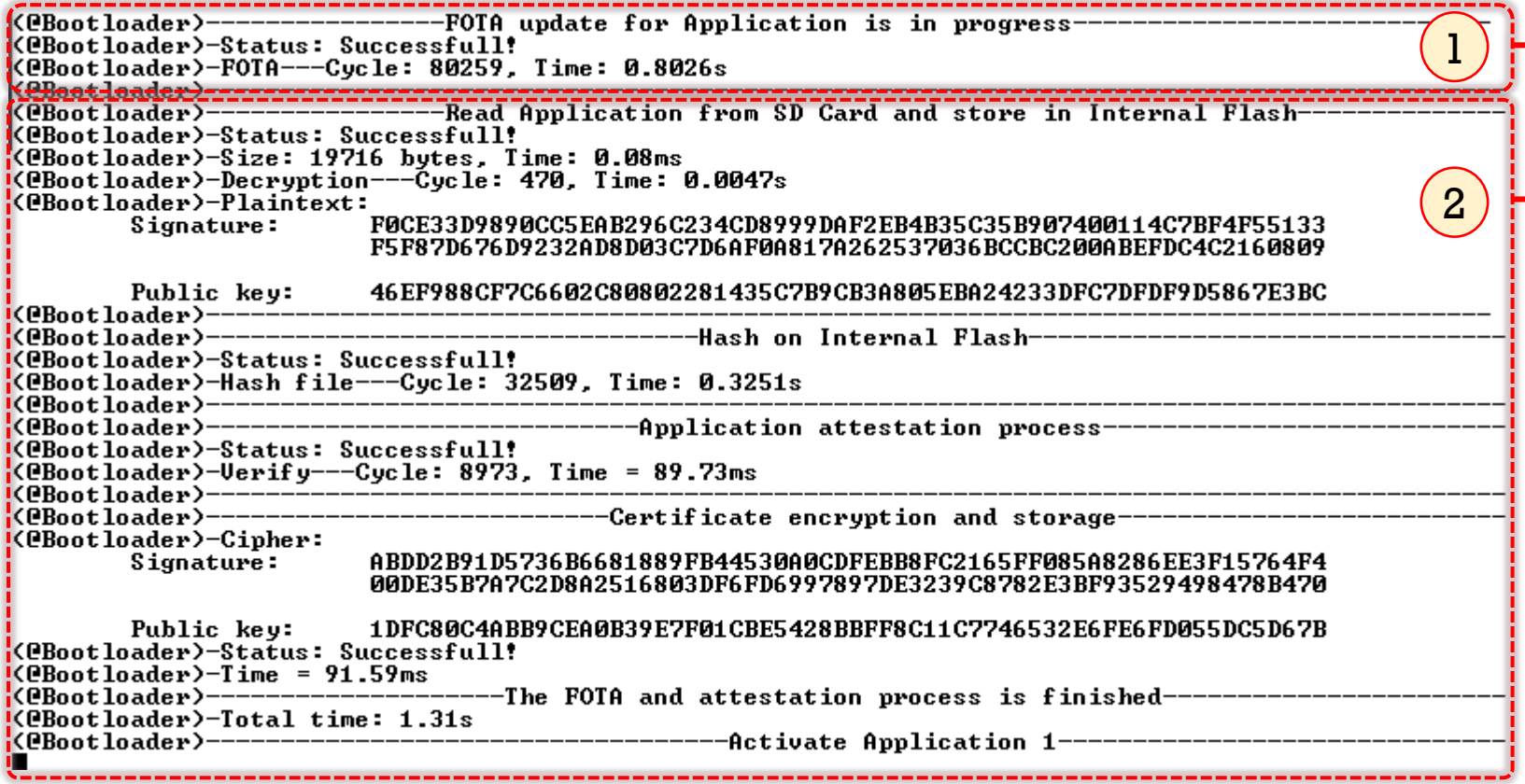
1. RoT authentication of the Bootloader

2. Bootloader authentication of the Application

```
(@RoT)-------------------------------Start the RoT Boot process-------------------------------
(@RoT)-Bootloader version is running: 2
(@RoT)----------------------------Read Certificates from External Flash----------------------------
(@RoT)-Status: Successfull!
(@RoT)-Time: 5.60ms
(@RoT)-Cipher
        Signature:      6578373B5D8F90242F121489DFE13AD5CF6FED2EBD6417916D3E8A0DAECEF4F9
                        57005CF1F6F3A90A92C7E5303E8994E33562A32E6ADD027BFE03135F65982E77

        Public key:     5461C0D6FDE4983C6084F2FBF8071416E8F1E4352AAD4E825E1FAA17B9231E95
(@RoT)-----------------------------------------------------------------------------------------
(@RoT)---------------------------------Certificates decryption process---------------------------------
(@RoT)-Status: Successfull!
(@RoT)-Time: 0.30ms
(@RoT)-Plaintext:
        Signature:      3E6B2F7301F03EA81C0C4909575803C2C33F29D968607760C507A895AB6CC13E
                        A22614213CA343054EAC2139FE558CE18499A614C96990600D88C7D223F6920E

        Public key:     0F72D89EA19B36B0539AAF7B70BE2D01E4A120C2FFA92E73F626888FBC812B52
(@RoT)-----------------------------------------------------------------------------------------
(@RoT)-------------------------------------Hash on Internal Flash-------------------------------------
(@RoT)-Status: Successfull!
(@RoT)-Size: 127184 bytes,Time: 2.11s
(@RoT)-----------------------------------------------------------------------------------------
(@RoT)---------------------------------Bootloader attestation process---------------------------------
(@RoT)-Status: Successfull!
(@RoT)-Time = 90.20ms
(@RoT)-----------------------------------------------------------------------------------------
(@RoT)-Total time of RoT Boot process: 2.20s
(@RoT)-----------------------------The RoT Boot process is finished-----------------------------
(@RoT)-------------------------------------Activate Bootloader-------------------------------------
(@Bootloader)-------------------------Start the Bootloader Boot process-------------------------
(@Bootloader)-Application version is running: 2
(@Bootloader)----------------------------Read Certificates from External Flash----------------------------
(@Bootloader)-Time = 7.77ms
(@Bootloader)-Cipher:
        Signature:      1A7E0161E995E499C51AC07A90953F66447E14D1D149C210837311815B55DA6F
                        7B031C84DE0BA999C0651796E033A9D41D1F203060DE1984EC05707DF1BE397E

        Public key:     5DCDD3886DC9CD52BA237867833D1F62EBA82B45D1A7E33854AEC9E19417EEB2
(@Bootloader)-----------------------------------------------------------------------------------------
(@Bootloader)---------------------------------Certificates decryption process---------------------------------
(@Bootloader)-Status: Successfull!
(@Bootloader)-Time = 0.22ms
(@Bootloader)-Plaintext:
        Signature:      416D1929B5EA4A15F6049DFA182C0671482ED026044DA2E12B4A33195EF7EFA8
                        8E255454145B43961C0ED39F20EFB1D6ACE4250AC36A8B9F1F8EA4F0B7D08507

        Public key:     06DECBC031B663DE893D25E70B842675E7F8EFB204A383C9FC97EB7991B5DB75
(@Bootloader)-----------------------------------------------------------------------------------------
(@Bootloader)-------------------------------------Hash on Internal Flash-------------------------------------
(@Bootloader)-Status: Successfull!
(@Bootloader)-Size: 19716
(@Bootloader)-Hash file---Cycle: 32509, Time: 0.3251s
(@Bootloader)-----------------------------------------------------------------------------------------
(@Bootloader)---------------------------------Application attestation process---------------------------------
(@Bootloader)-Status: Successfull!
(@Bootloader)-Verify---Cycle: 9002, Time: 90.02ms
(@Bootloader)-----------------------------------------------------------------------------------------
(@Bootloader)-Total time of Bootloader Boot process: 0.42s
(@Bootloader)-----------------------------The Bootloader Boot process is finished-----------------------------
(@Bootloader)-------------------------------------Activate Application 2-------------------------------------
```

4. Implementation results

The FOTA process for the Application



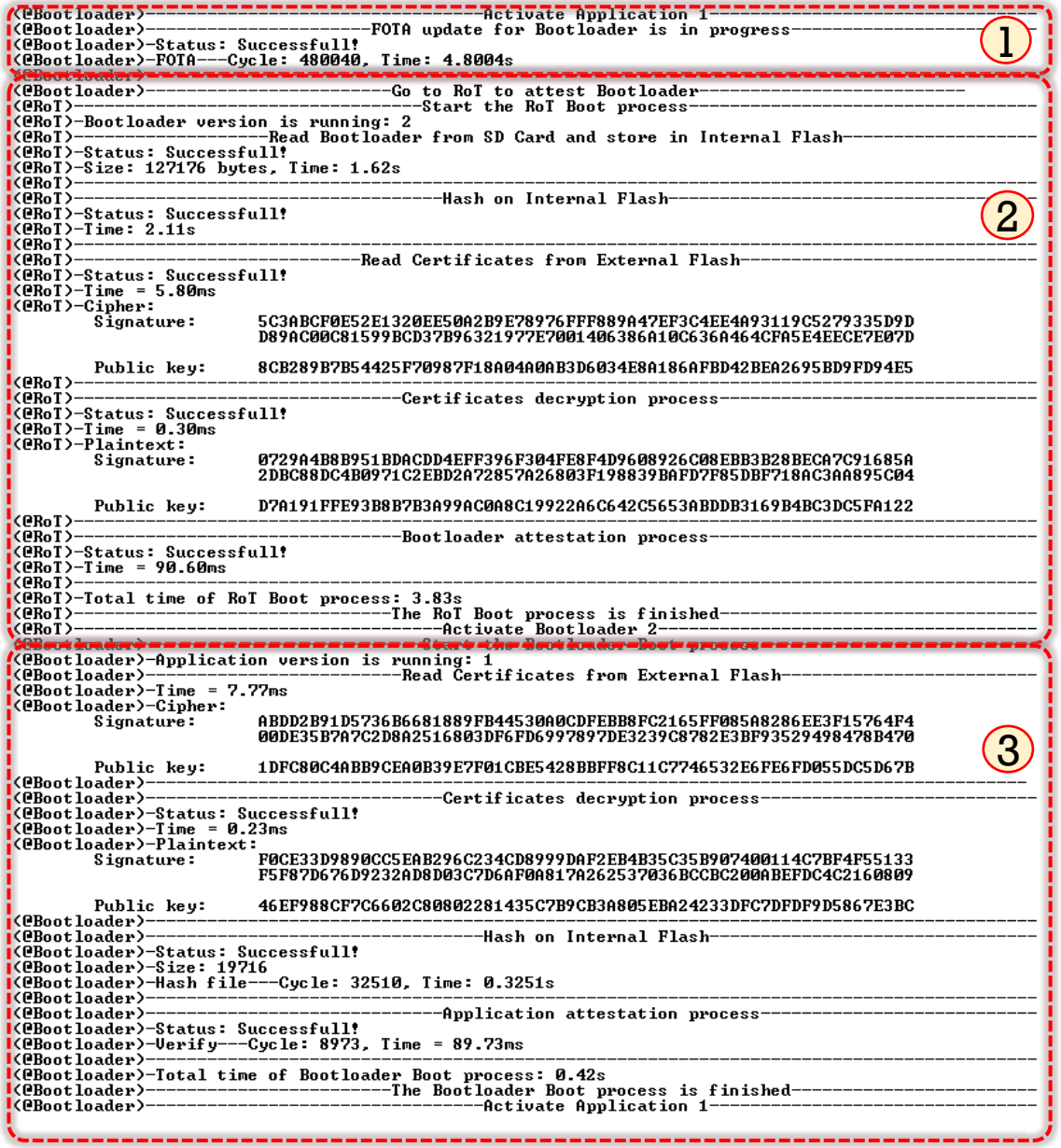
1 → The FOTA process

2 → The process of authenticating a new Application version

4. Implementation results

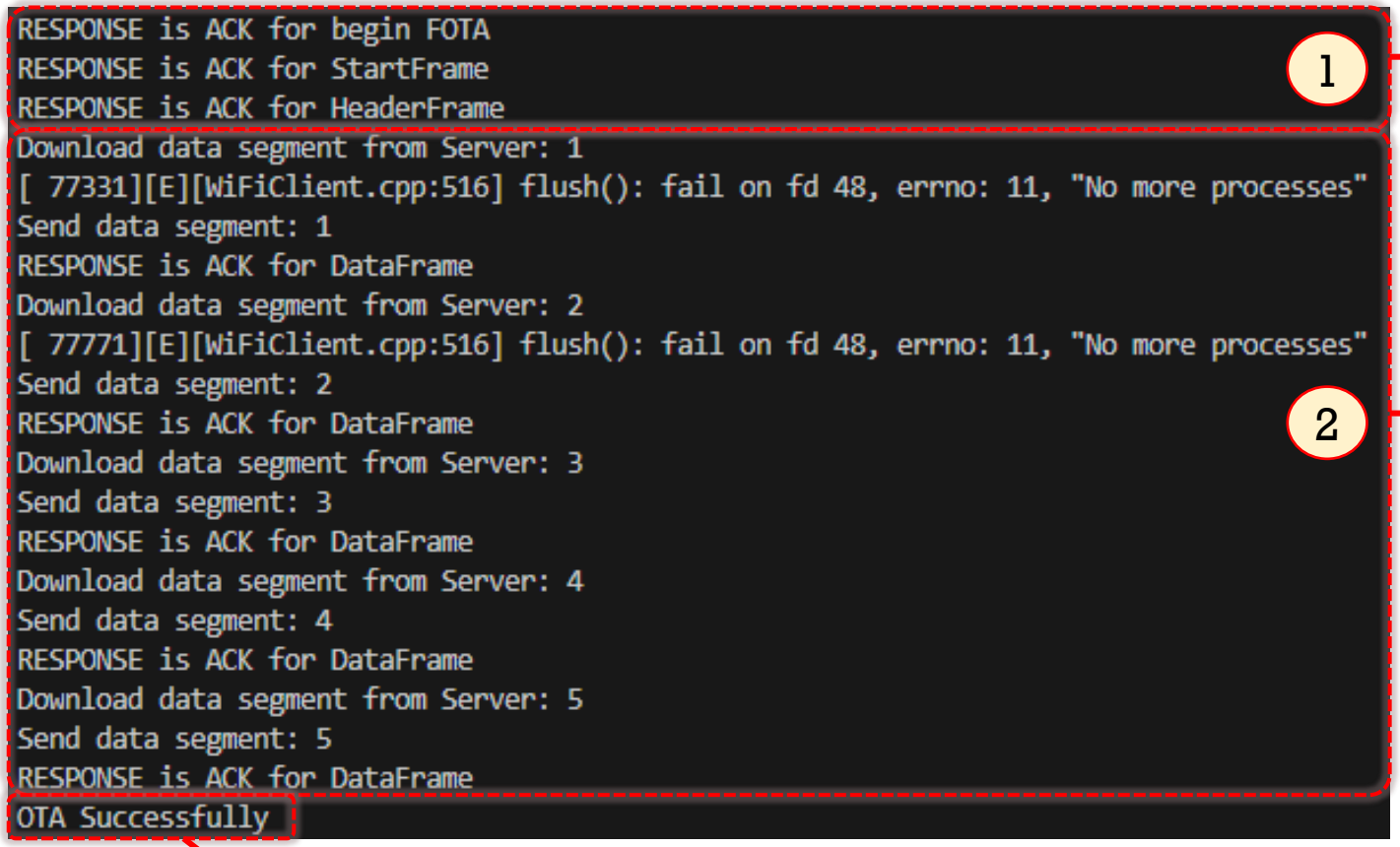
The FOTA process for the Bootloader

1. FOTA Process
2. The process of RoT authenticating a new Bootloader
3. The process of a new Bootloader authenticating an Application



4. Implementation results

The FOTA process on the ESP32 side



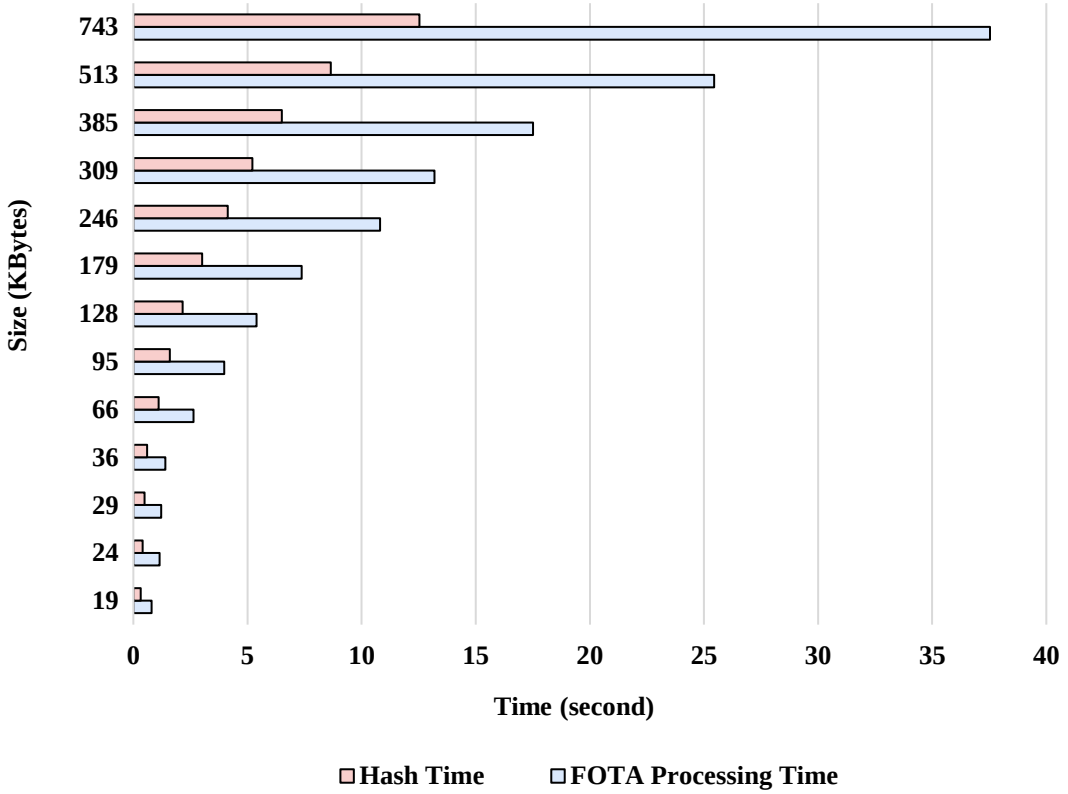
1 Sending a start signal and information for the FOTA process

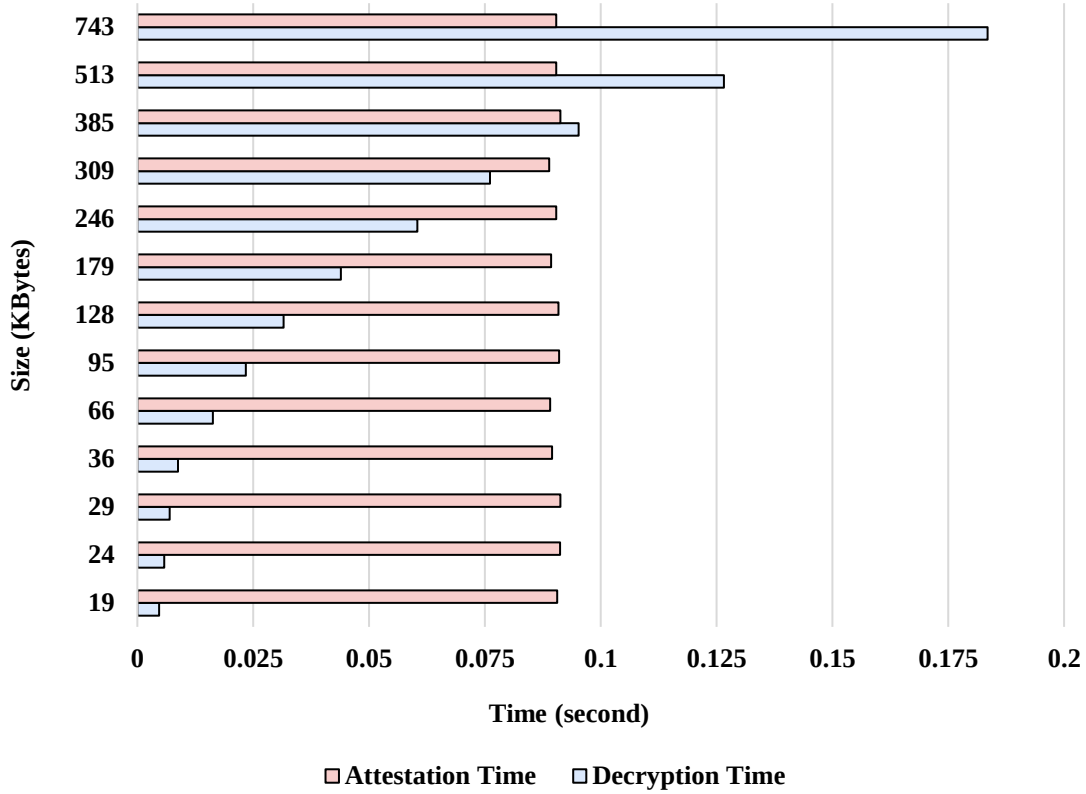
2 Downloading the file and sending it to the STM32

3 Receiving update status reports sent from STM32

4. Implementation results

FOTA PROCESSING – DATA COLLECTION





MCU Clock = 168MHz, UART baudrate = 912500bps

4. Implementation results

FOTA PROCESSING

	"Firmware Over-The-Air (FOTA) system in this project
Data Encryption Algorithm	ChaCha20
Digital Signature Algorithm	ED25519 combine SHA3
File Transfer Protocol between controllers/nodes	UART
FOTA processing time (1 Node)	Approximately 37.53 seconds (File 743 kilobytes)
Time to authenticate new firmware	Approximately 0.09 seconds (almost constant)
Time to decrypt new firmware	Approximately 0.1835 seconds (File 743 kilobytes)

4. Implementation results

Demo

Command Prompt

```
C:\Embedded_System\KLTN\CODE\DEVC\ED25519_SHA3>ED25519_SHA3.exe "C:\Users\Admin\STM32CubeIDE\RoadToBootloader\BOOTLOADER_ED25519_SHA3_new\Debug\BOOTLOADER_ED25519_SHA3_new.bin" "C:\Embedded_System\File_Folder\BOOTLOADER_F407_SIGNED_ENCRYPTED.bin" BOOTLOADER
```

5. Conclusion and Future Directions

The achieved results

- Successfully building an FOTA system for an IoT network with integrated security for fast and accurate results.
- Implementing, executing, and testing the feasibility of the ED25519 digital signature generation algorithms, SHA3 hash function, ChaCha20 data encryption algorithm, and Tree Hash algorithm for hashing large files when deployed on ARM core controllers.

5. Conclusion and Future Directions

Limitations

- ❑ Not yet implemented FOTA for an IoT network.
- ❑ Energy consumption of the system has not been tested yet.

5. Conclusion and Future Directions

Development direction.

- ❑ Deploying on multi-core microcontrollers.
- ❑ Building on FPGA.
- ❑ Implementing in an IoT network.
- ❑ Enhancing the system for stricter error prevention and handling.

1. Mahfoudhi, F., Sultania, A. K., Famaey, J.: Over-the-Air Firmware Updates for Constrained NB-IoT Devices. Sensors, 22, 7572 (2022).

2. Anastasiou, A., Christodoulou, P., Christodoulou, K., Vassiliou, V., Zinonos, Z.: IoT Device Firmware Update over LoRa: The Blockchain Solution. In: 16th International Conference on Distributed Computing in Sensor Systems (DCOSS), pp. 404-411. IEEE, Marina del Rey, CA, USA (2020).

[14] Vahdati, Z., Yasin, S., Ghasempour, A., and Salehi, M., “Comparison of ECC and RSA algorithms in IoT devices,” Journal of Theoretical and Applied Information Technology 97, 16 (2019).

[15] Suárez-Albela, M., et al. “A Practical Performance Comparison of ECC and RSA for Resource-Constrained IoT Devices,” 2018 Global Internet of Things Summit (GIoTS), Bilbao, Spain, pp. 1-6 (2018).

[16] Al Shaikhli, I., Alahmad, M., Munthir, K. “Hash function of finalist SHA-3: Analysis study,” International Journal of Advanced Computer Science and Information Technology (IJACSIT) Vol, 2, 1-12 (2014).

[17] Sarker, V. K., Gia, T. N., Tenhunen, H., Westerlund, T., "Lightweight Security Algorithms for Resource-constrained IoT-based Sensor Nodes," ICC 2020 - 2020 IEEE International Conference on Communications (ICC), Dublin, Ireland, pp. 1-7 (2020).

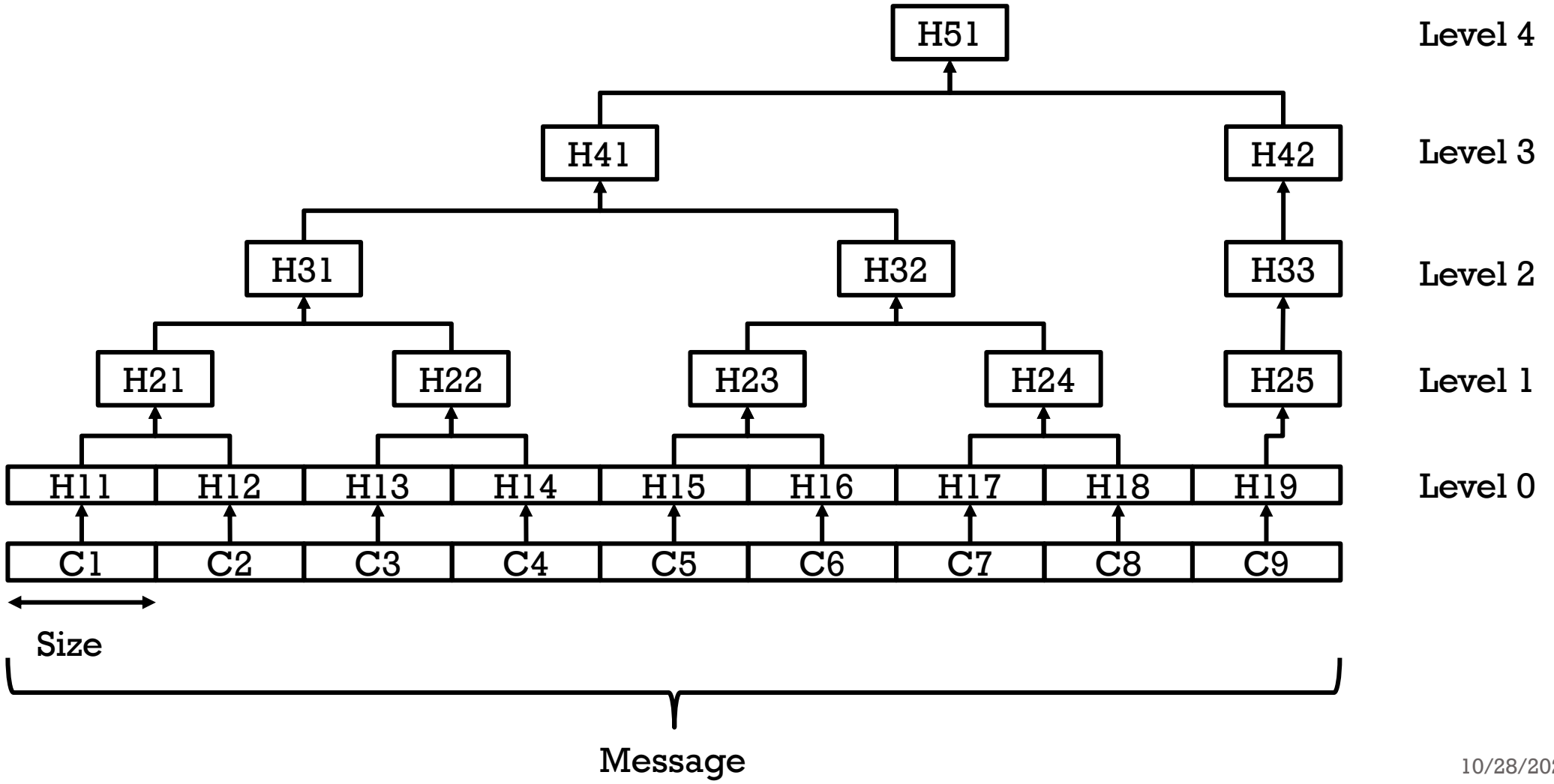
[18] Lachner, C., Dustdar, S., "A Performance Evaluation of Data Protection Mechanisms for Resource Constrained IoT Devices," 2019 IEEE International Conference on Fog Computing (ICFC), Prague, Czech Republic, pp. 47-52 (2019).

THE PRESENTATION PART ENDS HERE

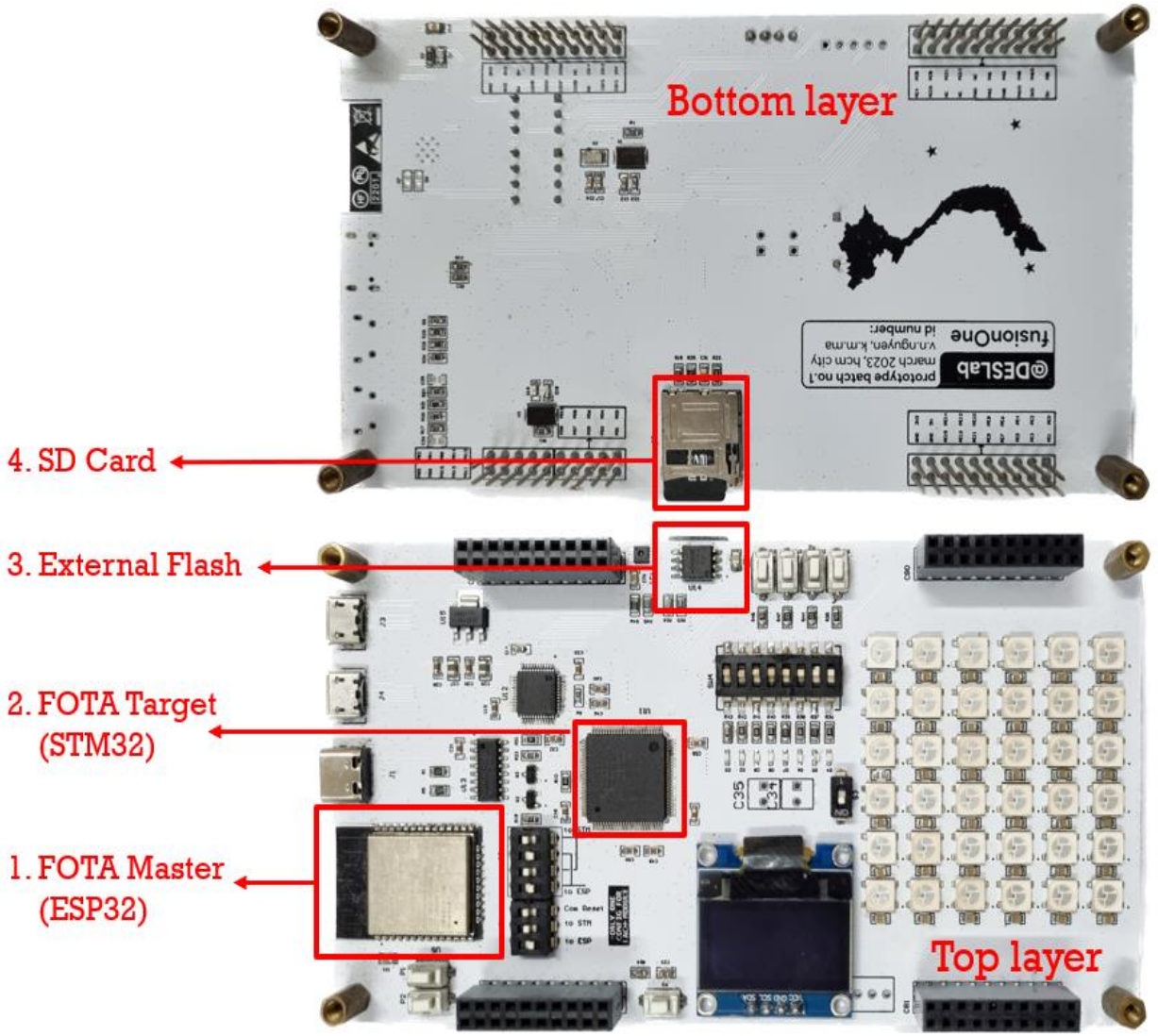
THANKS FOR WATCHING

ADDITIONAL RESULT

Tree Hash



RESULTS


Bottom layer
4. SD Card
3. External Flash
2. FOTA Target
(STM32)
1. FOTA Master
(ESP32)
Top layer

RESULTS

FOTA Command	SOF	Command	1	FOTA Start/Stop	CRC32	EOF
FOTA Header	SOF	Header	16	Header content	CRC32	EOF
FOTA Data	SOF	Data	N	Data content	CRC32	EOF
FOTA Response	SOF	Response	1	ACK/NACK	CRC32	EOF