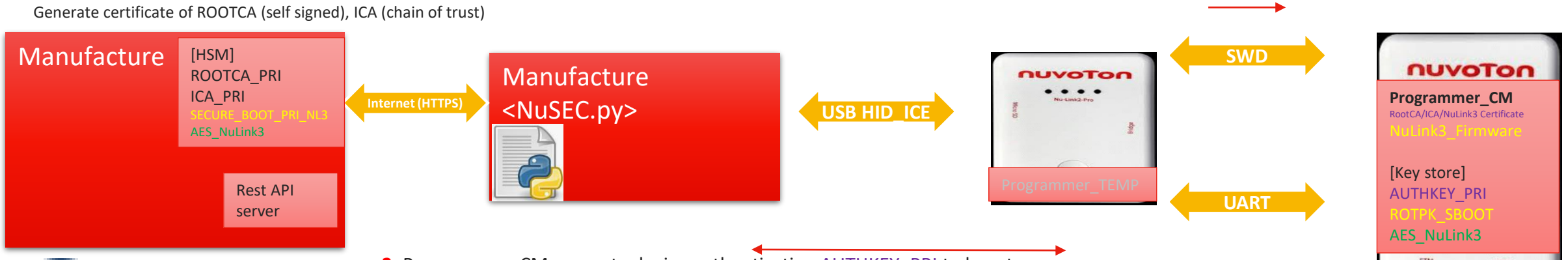
Manufacture stage

Prepare NuLink3 programmer for CM (In an OEM's secure environment)

Cloud Server | Desktop PC | Adapter | target board

1. OEM setup Rest API server
Generate private key for ROOTCA, ICA in HSM
Generate certificate of ROOTCA (self signed), ICA (chain of trust)

2. OEM get a blank NuLink3 as Programmer_CM
Write DevAuth.bin to SRAM of Programmer_CM (SWD)
Run DevAuth.bin and Programmer_CM talk with host



3. Programmer_CM generate device authentication AUTHKEY_PRI to key store
CLOUD sign Programmer_CM public key by ICA_PRI and burn RootCA/ICA/NuLink3 certificate to Programmer_CM

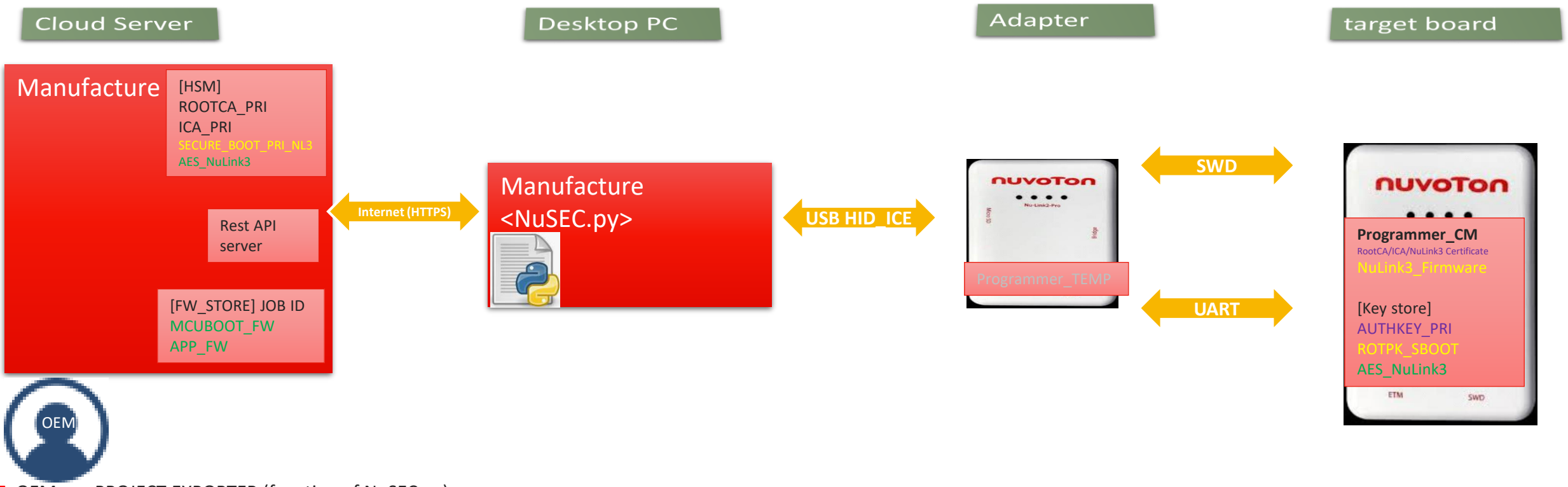
The RootCA is owned by OEM
ICAs belong to different projects of OEM

4. NuSEC.py ask CLOUD server to generate SECURE_BOOT_PRI_NL3 for Programmer_CM
NuSEC.py get SECURE_BOOT public key from CLOUD and burn to Programmer_CM ROTPK_SBOOT
NuSEC.py send NuLink3 firmware to cloud server and HSM signed it by SECURE_BOOT_PRI_NL3
NuSEC.py install NuLink3 firmware (signed) to Programmer_CM
Enable secure lock of Programmer_CM (Restricting debugger access)

5. Run NuLink3 firmware, Generate AES_NuLink3 by ECDH between HSM and Programmer_CM

6. OEM send Programmer_CM adapter to CM as secure programmer

EXPORT PROJECT for CM (by OEM)



7. OEM use PROJECT EXPORTER (function of NuSEC.py)
   Generating a package for the CM, the package include:
- Production count
- JOB ID (AS A TOKEN)
- IDs to check (e.g. UID list provided by chip vendor)

8. Encrypt package by AES_NuLink3, OEM send encrypted package to CM
   OEM upload target firmware (MCUBOOT: BL2, APP_FW: BL3) to FW_STORE server
   Bind uploaded firmware with corresponding Job ID

9. OEM send encrypted package to CM by Email

Import OEM package (CM)

Cloud Server | Desktop PC | Adapter (NuLink3) | target board

0. CM use package importer function of NuSEC.py to import OEM package to NuLink3

1. The package is processed in NuLink3 SECURE ENVIRONMENT to ensure authenticity and integrity after transporting from OEM to NuLink3.
The package is then decrypted (by AES_NuLink3) in NuLink3 SECURE ENVIRONMENT and production_count/Job_ID/.. are programmed in the NuLink3

Manufacture

[HSM]
ROOTCA_PRI
ICA_PRI
SECURE_BOOT_PRI_NL3
AES_NuLink3

[FW_STORE] JOB ID
MCUBOOT_FW
APP_FW

CSR (HTTPS)

Device Certificate (HTTPS)

Manufacturer
< NuSEC.py>

UART

2. Now NuLink3 is ready for the production job

nuvoTon
Programmer_CM
RootCA/ICA/NuLink3 Certificate
NuLink3_Firmware
JOB ID
Production_count

[Key store]
AUTHKEY_PRI
ROTPK_SBOOT
AES_NuLink3

ETM SWD

SWD
UART

MCU UID_A

MCU UID_B

3. For each provision.
NuLink3 decrease the production count number
Read device UID and communicate with cloud server
Device authentication provision (the following slide)
Secure boot key provision and FW installation (the following slide)
Firmware installation will be rejected if device authentication fail

Device authentication key provision

Cloud Server | Adapter (NuLink3) | target board

0. Trigger Mass Production flow by NuSEC.py, WIFI connect to Programmer_CM and it's the **start point** of target device provisioning procedure

1. Programmer_CM "mutual TLS" with Cloud, from authentication to get SESSION KEY
https://builtin.com/sites/www.builtin.com/files/styles/ckeditor_optimize/public/inline-images/2_mutual-tls-tutorial.jpg

2. *Program NuLink3 Certificate, ROOTCA CERT (SWD)*
Program DevAuth_MCU.bin to SRAM and run it (SWD)

3. Generate private key and store it in key store
這部份可能要MP版本的RSA

4. Target MCU send "pubkey+UID" to NuLink3
5. NuLink3 send CSR hash to Target MCU
6. Target MCU sign the CSR hash with private key of NuLink3

7. Send CSR (certificate signing request) to CLOUD

8. CLOUD verify signature, create device certificate DEV_CERT (UID, pubkey, ICA sign), and then provision it to the MCU

Manufacture
- [HSM]
  ROOTCA_PRI
  ICA_PRI
  SECURE_BOOT_PRI_NL3
  AES_NuLink3
- Rest API server
- [FW_STORE] JOB ID
  MCUBOOT_FW
  APP_FW

Ethernet (HTTPS) | WIFI AP | WIFI (HTTPS)

nuvoTon
Programmer_CM
RootCA/ICA/NuLink3 Certificate
NuLink3_Firmware
JOB ID
Production_count

[Key store]
AUTHKEY_PRI
ROTPK_SBOOT
AES_NuLink3

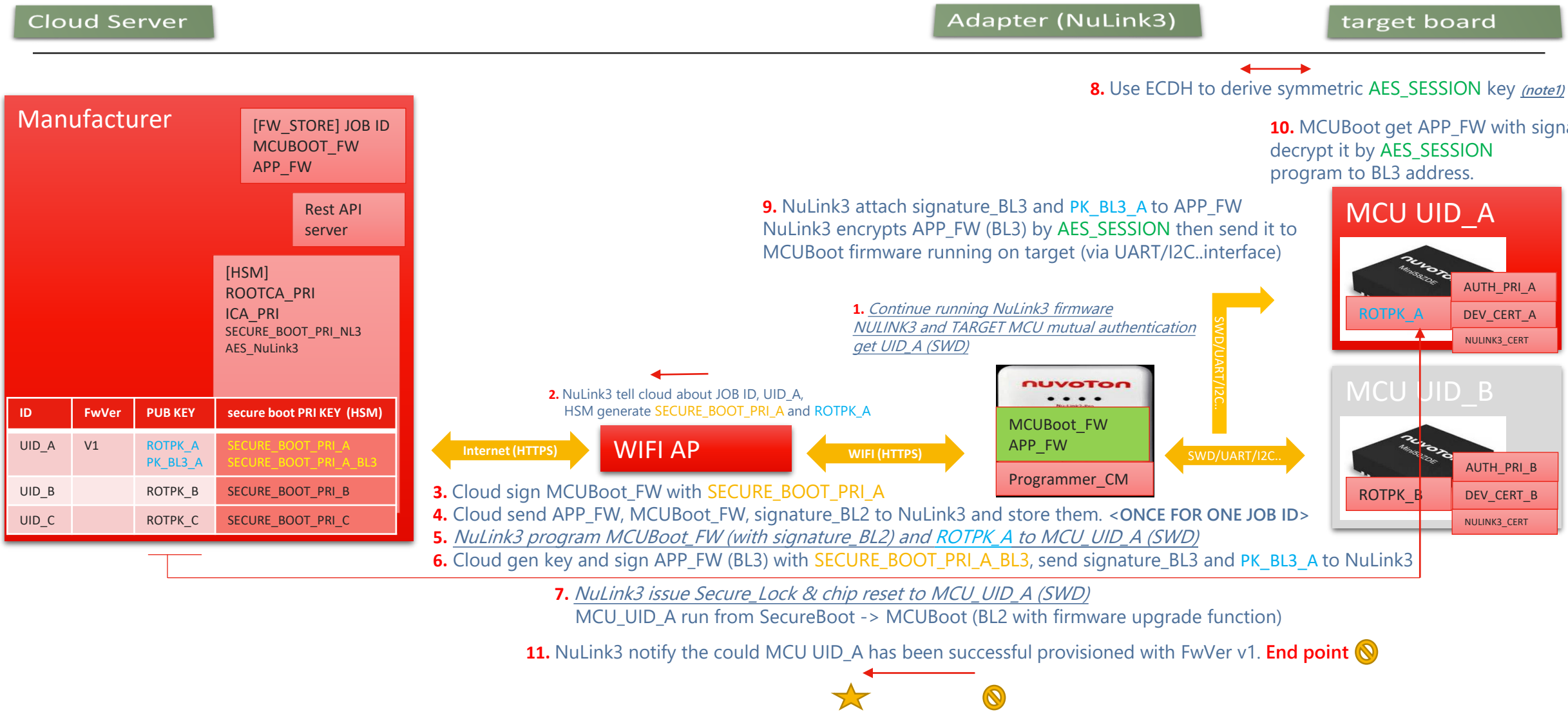
SWD/UART

MCU UID_A
- AUTH_PRI_A
- DEV_CERT_A
- NULINK3_CERT
- RootCA CERT

MCU UID_B
- AUTH_PRI_B
- DEV_CERT_B
- NULINK3_CERT
- RootCA CERT

Public key: 512 bits (64B)
Unique ID: 128 bits (16B)
Certificate: 16000 bits (2000bytes)

Firmware attestation - secure boot key and FW install



Status report

Cloud Server | Desktop PC | Adapter (NuLink3) | target board

Manufacture

[HSM]
ROOTCA_PRI
ICA_PRI
SECURE_BOOT_PRI_NL3
AES_NuLink3

[FW_STORE] JOB ID
MCUBOOT_FW
APP_FW

ID	FwVer	PUB KEY	secure boot PRI KEY (HSM)
UID_A	V1	ROTPK_A PK_BL3_A	SECURE_BOOT_PRI_A SECURE_BOOT_PRI_A_BL3
UID_B	v1	ROTPK_B	SECURE_BOOT_PRI_B
UID_C	v1	ROTPK_C	SECURE_BOOT_PRI_C

Internet (HTTPS)

Manufacturer
< NuSEC.py>

1. OEM/CM use NuSEC.py to dump the table of provision status, and firmware version of each device

SWD

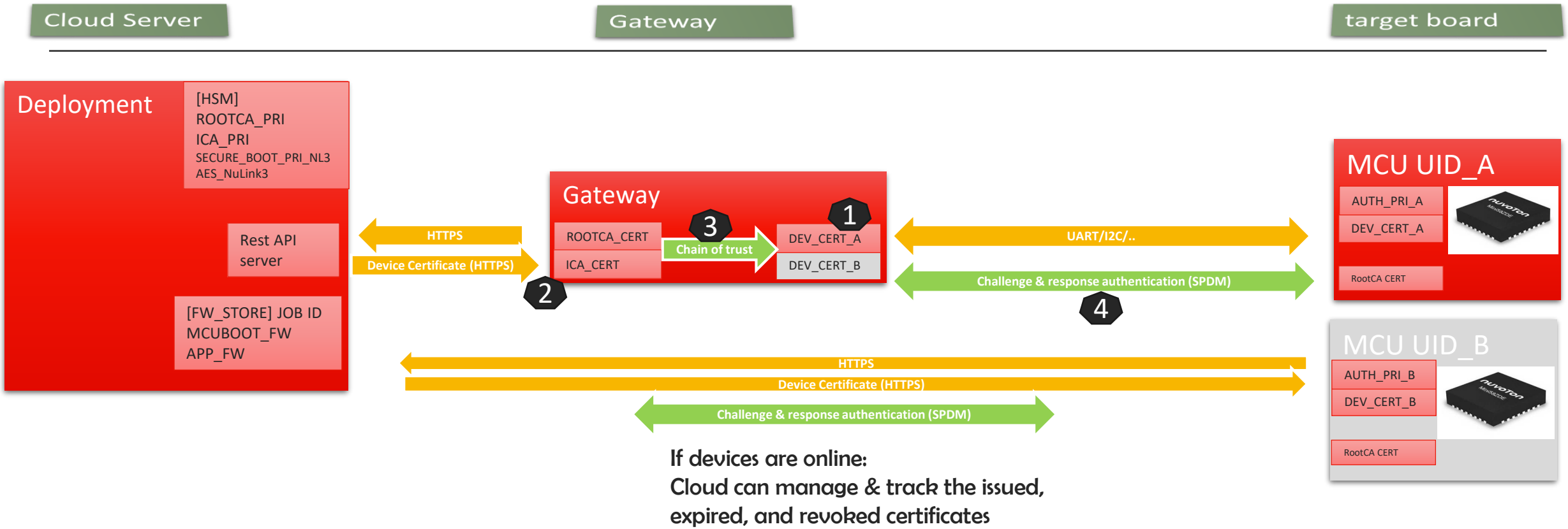
UART

MCU UID_A

MCU UID_B

Deployment stage

Device authentication



If devices are online:
Cloud can manage & track the issued, expired, and revoked certificates

Maintenance stage

Firmware upgrade (via NuLink3 bridge)

Cloud Server | Desktop PC | Adapter (NuLink3) | target board

4. Use ECDH to derive symmetric AES_SESSION key

5.
NuLink3 and Target MCU mutual authenticate (by DEV_CERT/ROOTCA/ICA cert, NULINK3_CERT)
NuLink3 check firmware rollback
NuLink3 attach signature_BL3 and PK_BL3_A to APP_FW2
NuLink3 encrypts APP_FW2 by AES_SESSION then send it to MCUBoot firmware running on target (via UART/I2C..interface)

6. MCUBoot get APP_FW with signature decrypt it by AES_SESSION program to BL3 address.

Manufacturer

[FW_STORE] JOB ID
MCUBOOT_FW
APP_FW2

Rest API server

[HSM]
ROOTCA_PRI
ICA_PRI

SECURE_BOOT_PRI_NL3
AES_NuLink3

ID	FwVer	PUB KEY	secure boot PRI KEY (HSM)
UID_A	V2 +1 after complete	ROTPK_A PK_BL3_A	SECURE_BOOT_PRI_A SECURE_BOOT_PRI_A_BL3
UID_B	V1	ROTPK_B	SECURE_BOOT_PRI_B
UID_C	V1	ROTPK_C	SECURE_BOOT_PRI_C

0. Put upgrade **APP_FW2** to FW_STORE server

1. Trigger FW upgrade,
NuLink3 run FW upgrade application,
Programmer_CM "mutually_TLS" with cloud
get UID_A and firmware version from device

2. NuLink3 tell cloud about UID_A, cloud know UID_A FwVer is V1 and need update

Ethernet (HTTPS) | WIFI AP | WIFI (HTTPS)

nuvoTon
APP_FW2
Programmer_CM
RootCA CERT
[Key store]
AUTHKEY_PRI

SWD/UART/I2C..

MCU UID_A
ROTPK_A
AUTH_PRI_A
DEV_CERT_A
NULINK3_CERT

MCU UID_B
ROTPK_B
AUTH_PRI_B
DEV_CERT_B
NULINK3_CERT

3. Cloud send APP_FW2 <**ONCE, if batch update**>
Cloud sign **APP_FW2** with SECURE_BOOT_PRI_A_BL3, send signature_BL3 and PK_BL3_A to NuLink3.

Internet (HTTPS)

Target MCU C

Firmware OTA upgrade (direct)

Cloud Server | Desktop PC | Adapter (NuLink3) | target board

Manufacturer

[FW_STORE] JOB ID
MCUBOOT_FW
APP_FW2

Rest API server

[HSM]
ROOTCA_PRI
ICA_PRI

SECURE_BOOT_PRI_NL3
AES_NuLink3

ID	PUB KEY	secure boot PRI KEY (HSM)
UID_A	ROTPK_A PK_BL3_A	SECURE_BOOT_PRI_A SECURE_BOOT_PRI_A_BL3
UID_B	ROTPK_B	SECURE_BOOT_PRI_B
UID_C	ROTPK_C	SECURE_BOOT_PRI_C

Put upgrade **APP_FW2** to FW_STORE server

MCUBoot (Signed)
APP_FW2

Programmer_CM

SWD/UART/I2C..

SWD/UART/I2C..

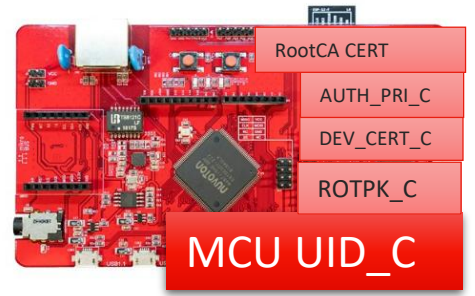
MCU UID_A

MCU UID_B

MCUBoot get APP_FW with signature,
decrypt it by AES_SESSION
program to BL3 address.

HTTPS/MQTTS

Device and cloud do mutual_TLS and derive AES_SESSION
Cloud prevent firmware rollback
Cloud attach signature_BL3 and PK_BL3_A to APP_FW2
Cloud encrypts APP_FW2 by AES_SESSION then send it to MCUBoot



Decommission

Decommission

Cloud Server

Adapter (NuLink3)

target board

Manufacturer

[FW_STORE] JOB ID
MCUBOOT_FW
APP_FW2

Rest API server

[HSM]
ROOTCA_PRI
ICA_PRI

SECURE_BOOT_PRI_NL3
AES_NuLink3

ID	Revoke cert
UID_A	
UID_B	revoked
UID_C	revoked

Internet (HTTPS)

WIFI AP

WIFI (HTTPS)

SWD/UART/I2C..

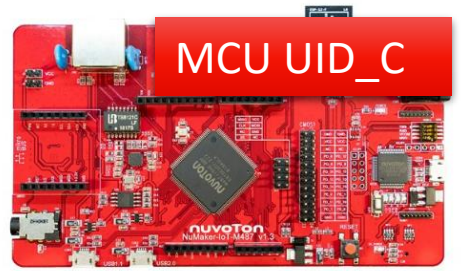
UART

MCU UID_A

MCU UID_B

HTTPS

Decommission device



MCU UID_C

Current status

- Current status (Coworking with A008 JY33)
  - Device authentication: AWS <-> NuSEC.py <-> NuLink2 <-> (M2354,KM1M7C)

Cloud Server | Desktop PC | Adapter | target board

1. Current we use AWS IoT to issue device certificate

Internet (HTTPS)

Manufacture
<NuSEC.py>

USB HID_ICE

2. Write DevAuth_MCU.bin to SRAM of device via SWD

SWD

UART

AUTHKEY_PRI
Certificate

Manufacture

[Soft HSM]
ROOTCA_PRI
ICA_PRI

SECURE_BOOT_PRI_NL3
AES_NuLink3

Rest API server

We setup server for next step
(use Soft HSM as HSM simulator)

3. Target MCU generate device authentication AUTHKEY_PRI to key store (UART command)
AWS sign Target MCU public key and burn certificate back to target MCU

UART

UART protocol of DevAuth_MCU.bin is standardized.
It can also communicates with other devices. (e.g. DATA IO)

Joy of innovation

NUVOTON

Thank You

Danke

Merci

ありがとう

Gracias

Kiitos

감사합니다

धन्यबाद

شكرا لك

הדות