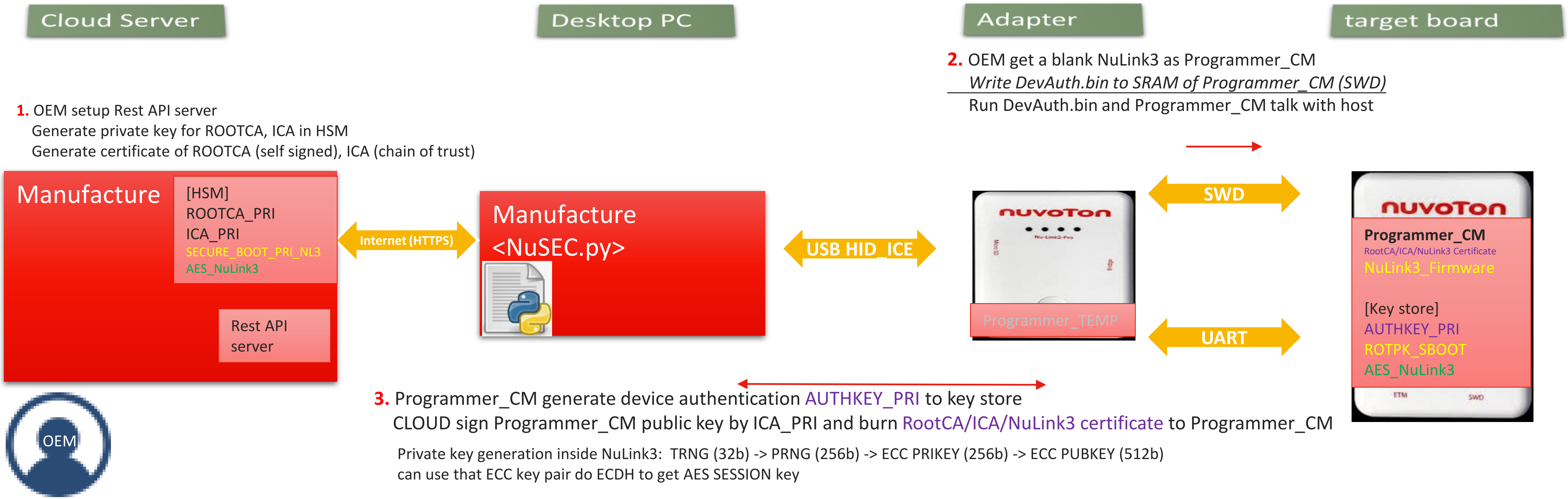
Manufacture stage

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



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

Private key generation inside NuLink3: TRNG (32b) -> PRNG (256b) -> ECC PRIKEY (256b) -> ECC PUBKEY (512b)
can use that ECC key pair do ECDH to get AES SESSION key

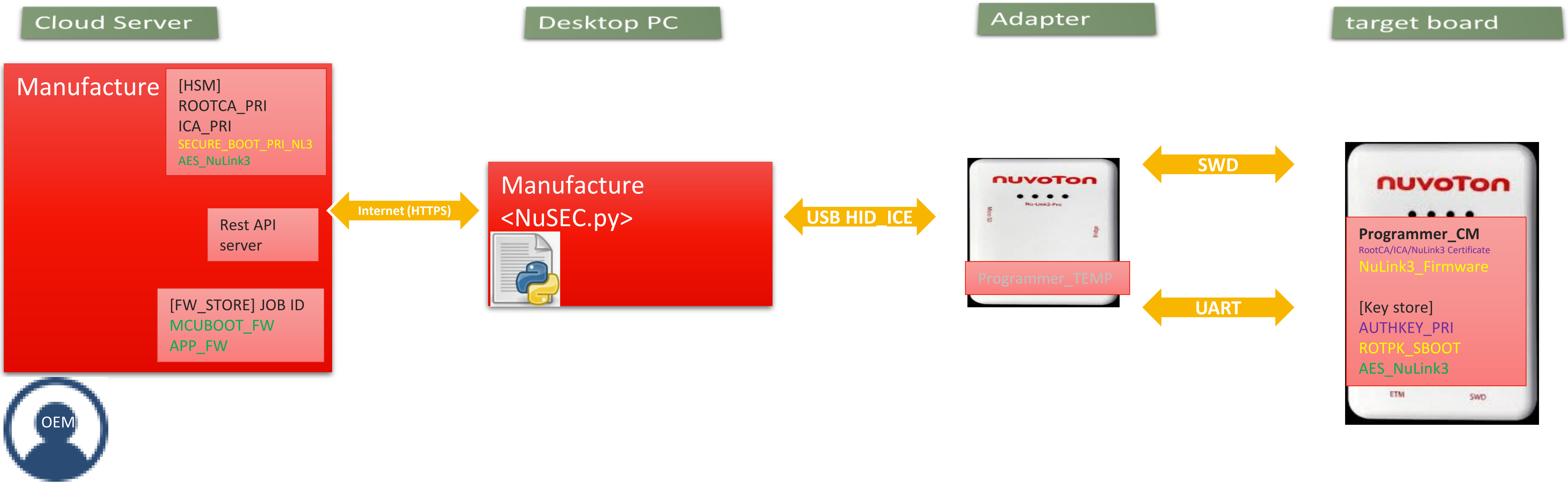
The RootCA is owned by OEM
ICAs belong to different projects of OEM

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)



Cloud Server

Manufacture

[HSM]
ROOTCA_PRI
ICA_PRI
SECURE_BOOT_PRI_NL3
AES_NuLink3

Rest API server

[FW_STORE] JOB ID
MCUBOOT_FW
APP_FW

Internet (HTTPS)

Desktop PC

Manufacture
<NuSEC.py>

USB HID_ICE

Adapter

Programmer_TEMP

SWD

UART

target board

Programmer_CM
RootCA/ICA/NuLink3 Certificate
NuLink3_Firmware

[Key store]
AUTHKEY_PRI
ROTPK_SBOOT
AES_NuLink3

7. OEM use PROJECT EXPORTER (function of NuSEC.py)
Generating a package for the CM, the package include:

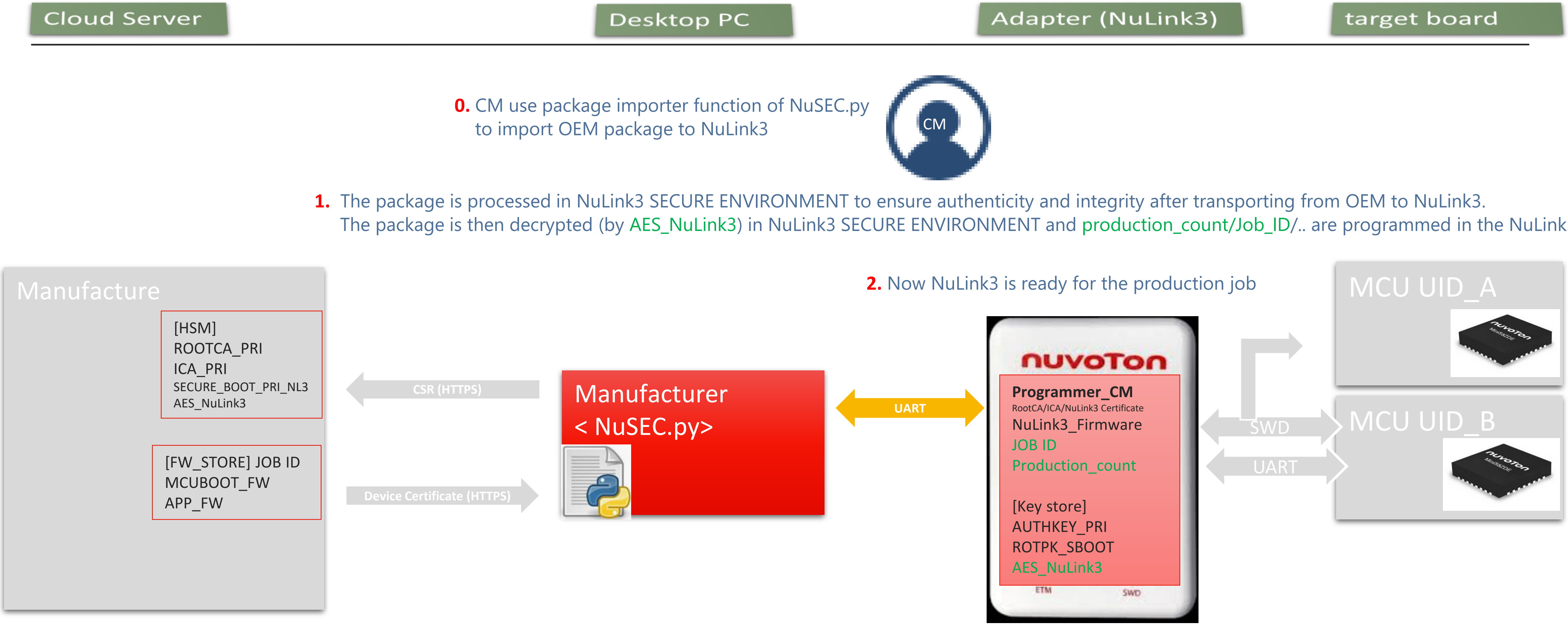
- Production count
- JOB ID (AS A TOKEN)
- IDs to check (e.g. UID list provided by chip vendor)

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

CM

9. OEM send encrypted package to CM by Email

Import OEM package (CM)



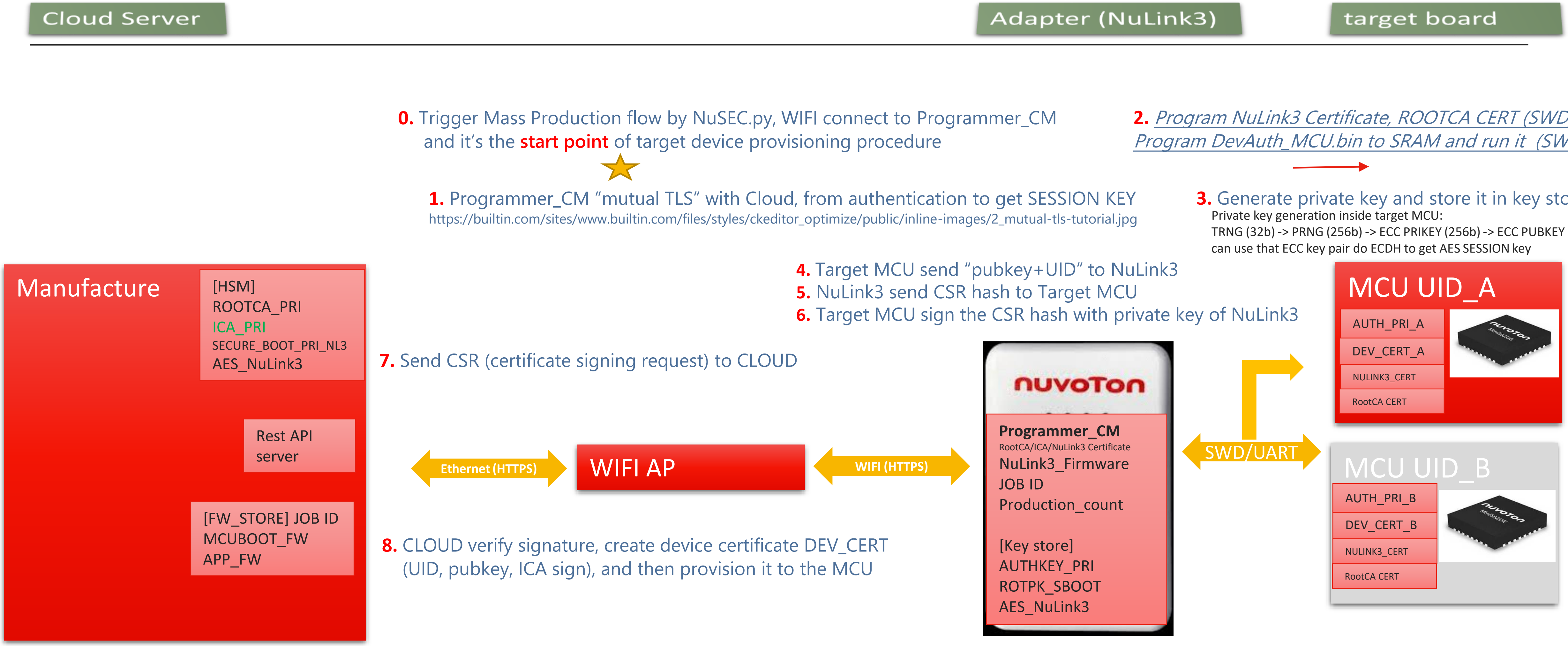
0. CM use package importer function of NuSEC.py to import OEM package to NuLink3

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

2. Now NuLink3 is ready for the production job

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
Private key generation inside target MCU:
TRNG (32b) -> PRNG (256b) -> ECC PRIKEY (256b) -> ECC PUBKEY (512b)
can use that ECC key pair do ECDH to get AES SESSION key

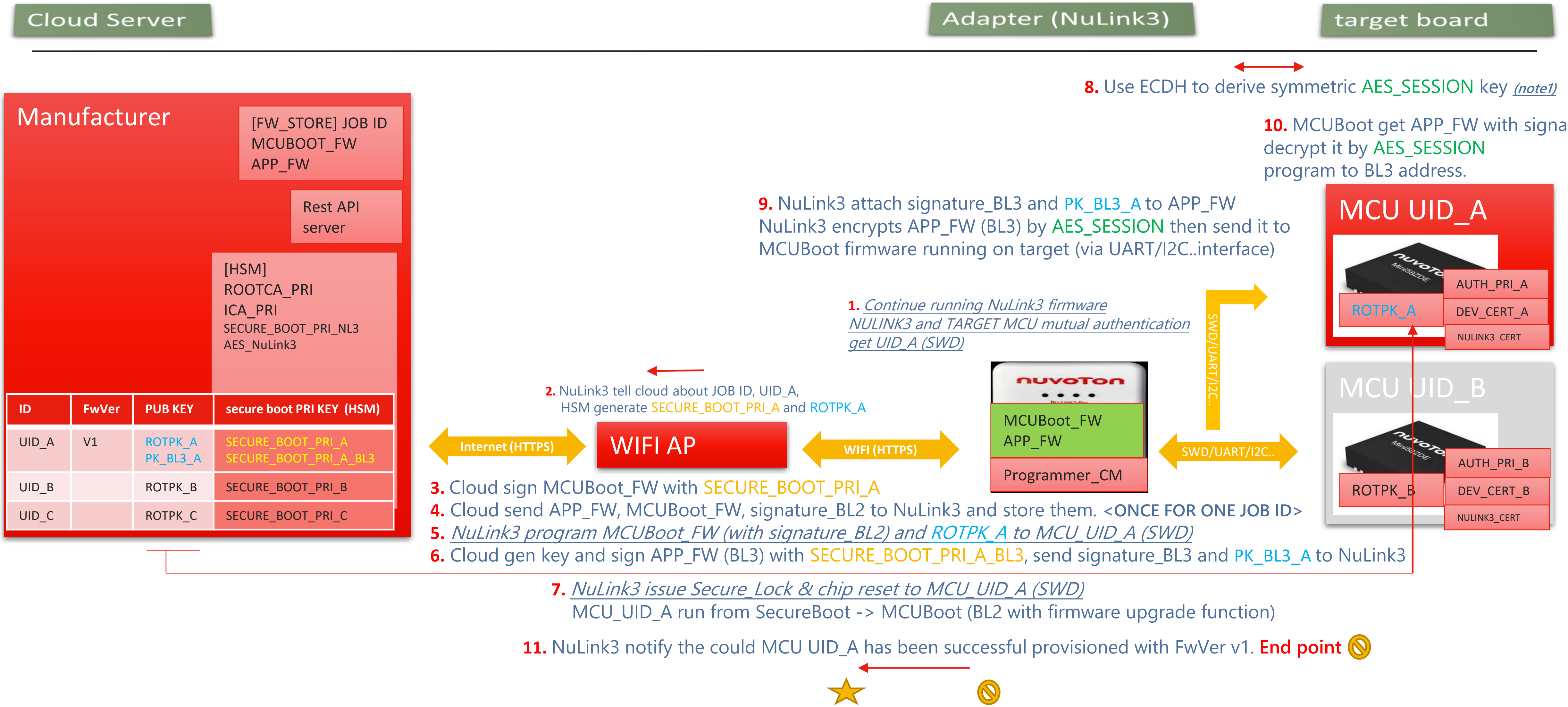
4. Target MCU send "pubkey+UID" to NuLink3
5. NuLink3 send CSR hash to Target MCU
6. Target MCU sign the CSR hash with private key of NuLink3

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

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

Firmware attestation - secure boot key and FW install



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		ROTPK_B	SECURE_BOOT_PRI_B
UID_C		ROTPK_C	SECURE_BOOT_PRI_C

1. *Continue running NuLink3 firmware*
NULINK3 and TARGET MCU mutual authentication
get UID_A (SWD)

2. NuLink3 tell cloud about JOB ID, UID_A,
HSM generate SECURE_BOOT_PRI_A and ROTPK_A

3. Cloud sign MCUBoot_FW with SECURE_BOOT_PRI_A

4. Cloud send APP_FW, MCUBoot_FW, signature_BL2 to NuLink3 and store them. <**ONCE FOR ONE JOB ID**>

5. *NuLink3 program MCUBoot_FW (with signature_BL2) and ROTPK_A to MCU_UID_A (SWD)*

6. Cloud gen key and sign APP_FW (BL3) with SECURE_BOOT_PRI_A_BL3, send signature_BL3 and PK_BL3_A to NuLink3

7. *NuLink3 issue Secure_Lock & chip reset to MCU_UID_A (SWD)*
MCU_UID_A run from SecureBoot -> MCUBoot (BL2 with firmware upgrade function)

8. Use ECDH to derive symmetric AES_SESSION key *(note1)*

9. NuLink3 attach signature_BL3 and PK_BL3_A to APP_FW
NuLink3 encrypts APP_FW (BL3) by AES_SESSION then send it to MCUBoot firmware running on target (via UART/I2C..interface)

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

11. NuLink3 notify the could MCU UID_A has been successful provisioned with FwVer v1. **End point** 🚫

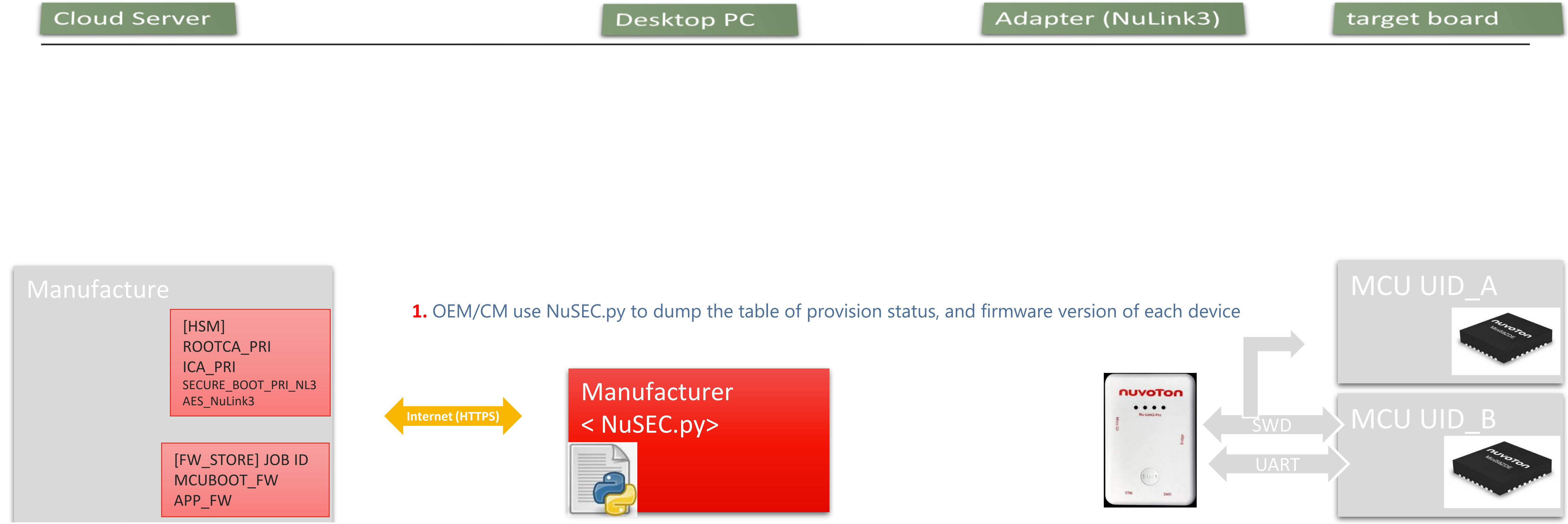
★ 🚫

12. Loop until production counter decrease to zero, each target MCU has being provisioned from **start point** to the **end point** by the NuLink3 system.

13. MCUBoot_FW and APP_FW on NuLink3 will be erased. Traceability log table can be found on the cloud

(note1) AES_SESSION may derive from AUTH_PRI_A and DEV_CERT_A

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

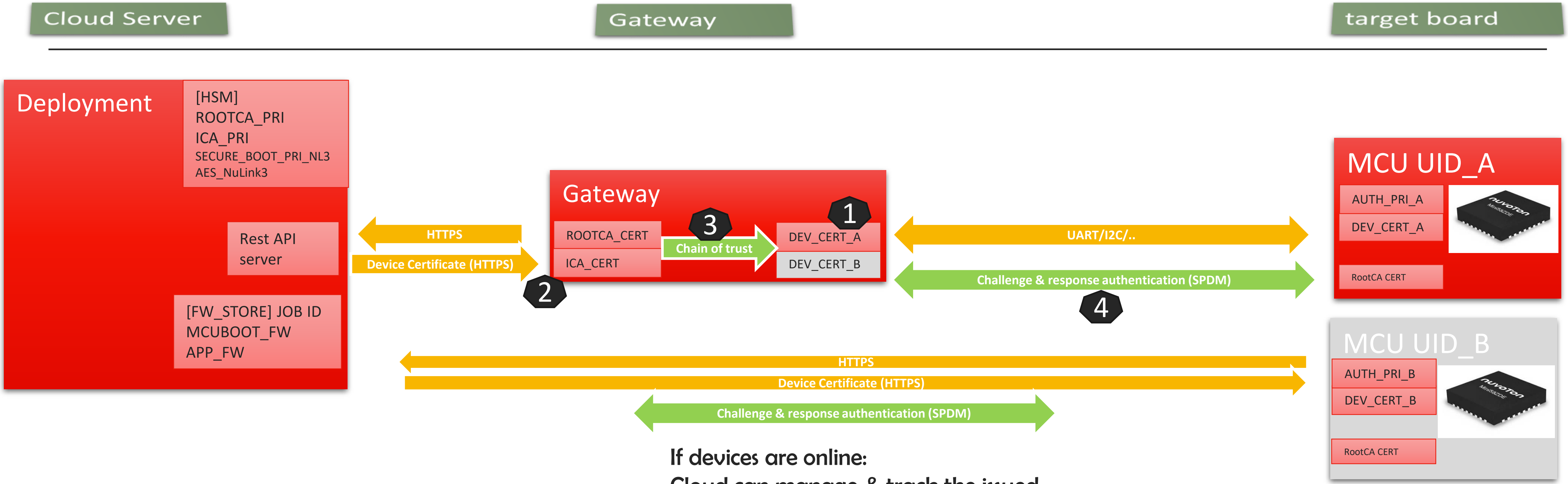
1. OEM/CM use NuSEC.py to dump the table of provision status, and firmware version of each device

Manufacturer
< NuSEC.py>

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

Deployment stage

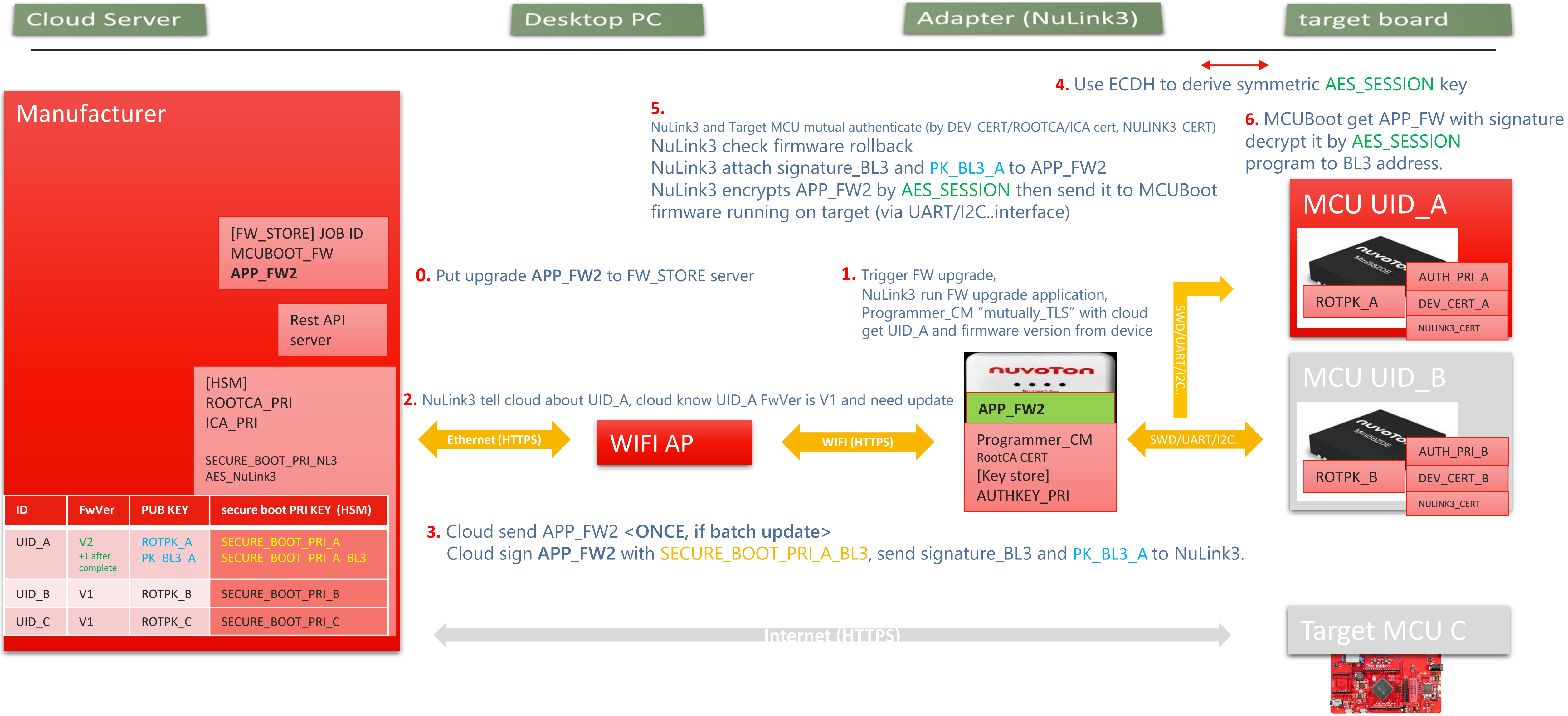
Device authentication



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

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

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.

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.

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

MCU UID_A

MCU UID_B

MCUBoot (Signed)
APP_FW2

Programmer_CM

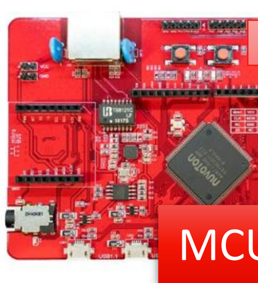
SWD/UART/I2C..

SWD/UART/I2C..

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

HTTPS/MQTTS

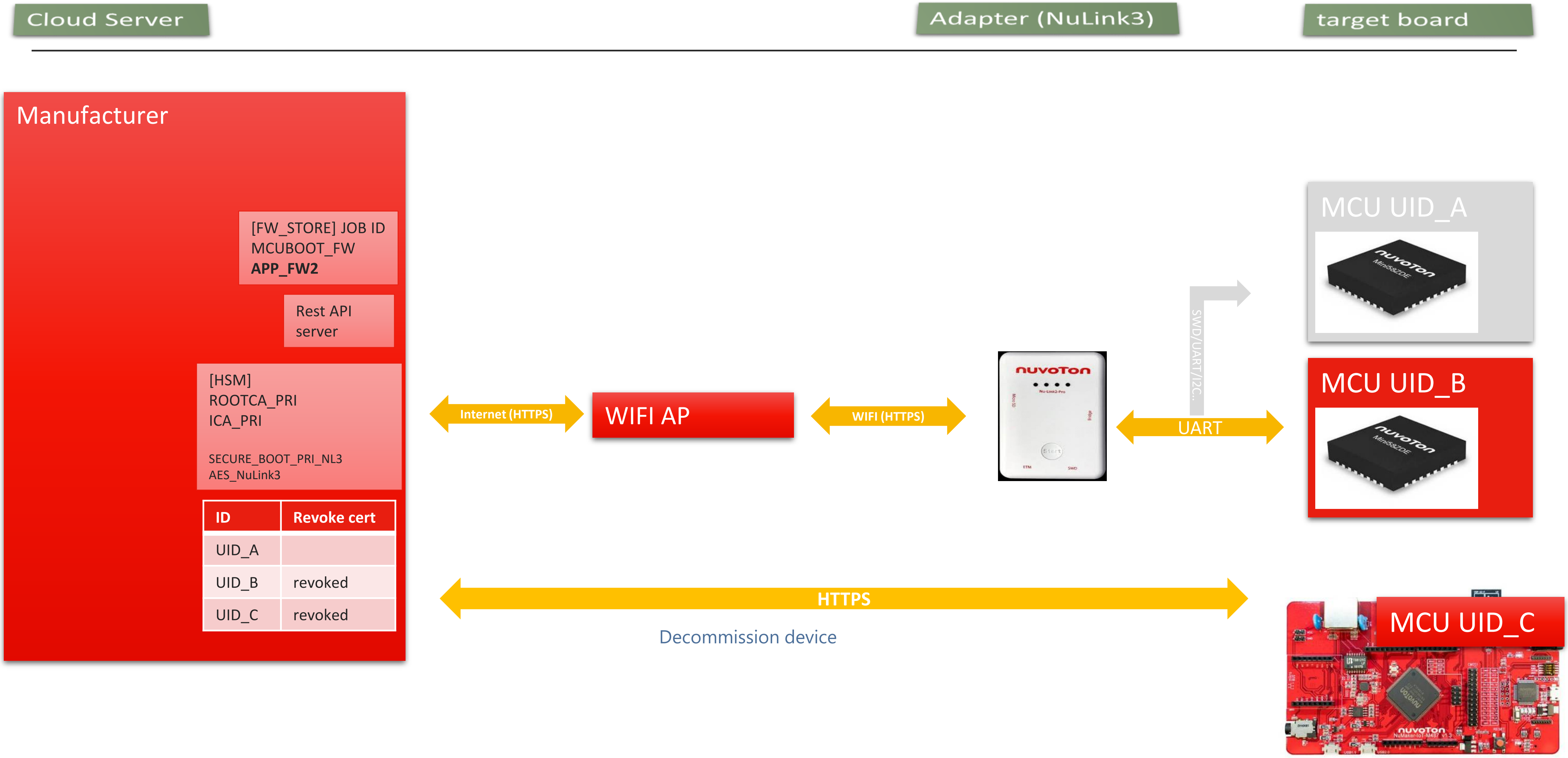
Device and cloud do mutual_TLS and derive AES_SESSION
Cloud prevent firmware rollback
Cloud attach signature_BL3 and PK_BL3_A to APP_FW2
Cloud encrypts APP_FW2 by AES_SESSION then send it to MCUBoot



RootCA CERT
AUTH_PRI_C
DEV_CERT_C
ROTPK_C
MCU UID_C

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

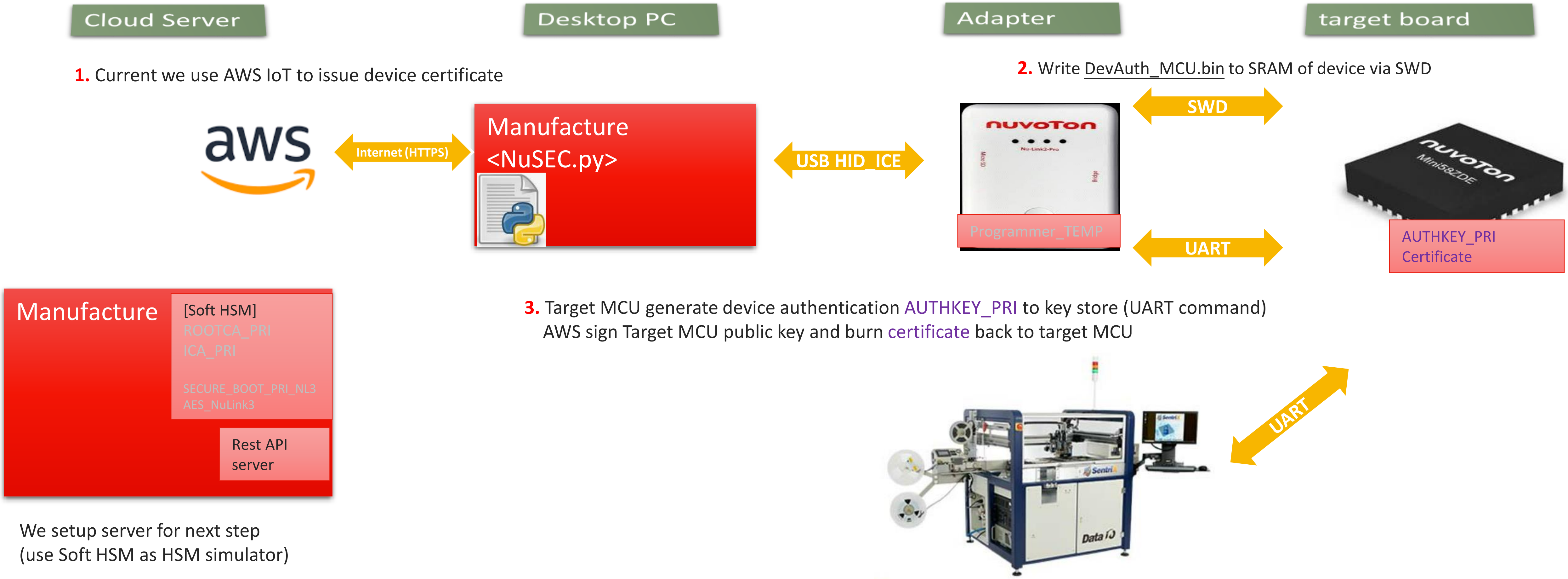
SECURE_BOOT_PRI_NL3
AES_NuLink3

ID	Revoke cert
UID_A	
UID_B	revoked
UID_C	revoked

Decommission device

Current status

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



1. Current we use AWS IoT to issue device certificate

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

Manufacture
<NuSEC.py>

AUTHKEY_PRI
Certificate

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

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)

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

لك اركش

הדות