

AppNote

-

Sigfox Credential provisioning for Open Software Architecture modules and SDK

Content

1	Document Purpose.....	3
2	Conditions to access to network.....	4
3	Sigfox credentials security.....	5
4	Credentials states.....	6
5	General purpose: Open architecture based module.....	7
5.1	Evaluation Provisioning with module and EVK (Evaluation kit).....	7
5.1.1	Open architecture module personalization.....	7
5.2	Provisioning and mass production with Open architecture module.....	11
5.2.1	Get access to your own credentials.....	11
5.2.2	Embedded device firmware without encrypted credentials.....	11
5.2.3	Embedded device firmware with encrypted credential.....	12
5.2.4	PC tool for decrypting credentials example.....	13
5.2.5	PC tool for decrypting re-encrypting credentials.....	13
5.2.6	Mass production manufacturing steps with Open architecture modules.....	14
6	Register/Activation End Device on Network.....	17
6.1	Activate prototypes and EVK credentials on the network.....	17
6.2	Activate mass production devices on network.....	18
7	Annex1: Use case Murata module CMWX1ZZABZ.....	19
7.1	Provisioning during evaluation and prototyping with samples.....	20
7.1.1	Prerequisites.....	21
7.1.2	Getting signature with EVK.....	21
7.1.3	Personalization of Murata samples and network activation.....	22
7.2	Mass production provisioning: custom personalization.....	23
7.2.1	Principle.....	24
7.2.2	ST SDK modifications for embedded device software	25
7.2.3	Flashing / personalization bench : PC software.....	26
7.2.4	How to manage Sigfox credentials for manufacturing.....	29
8	Annex 2: STEVAL EVK provisioning.....	32
8.1	STEVAL EVK evaluation personalization.....	32
8.2	Mass production personalization.....	33

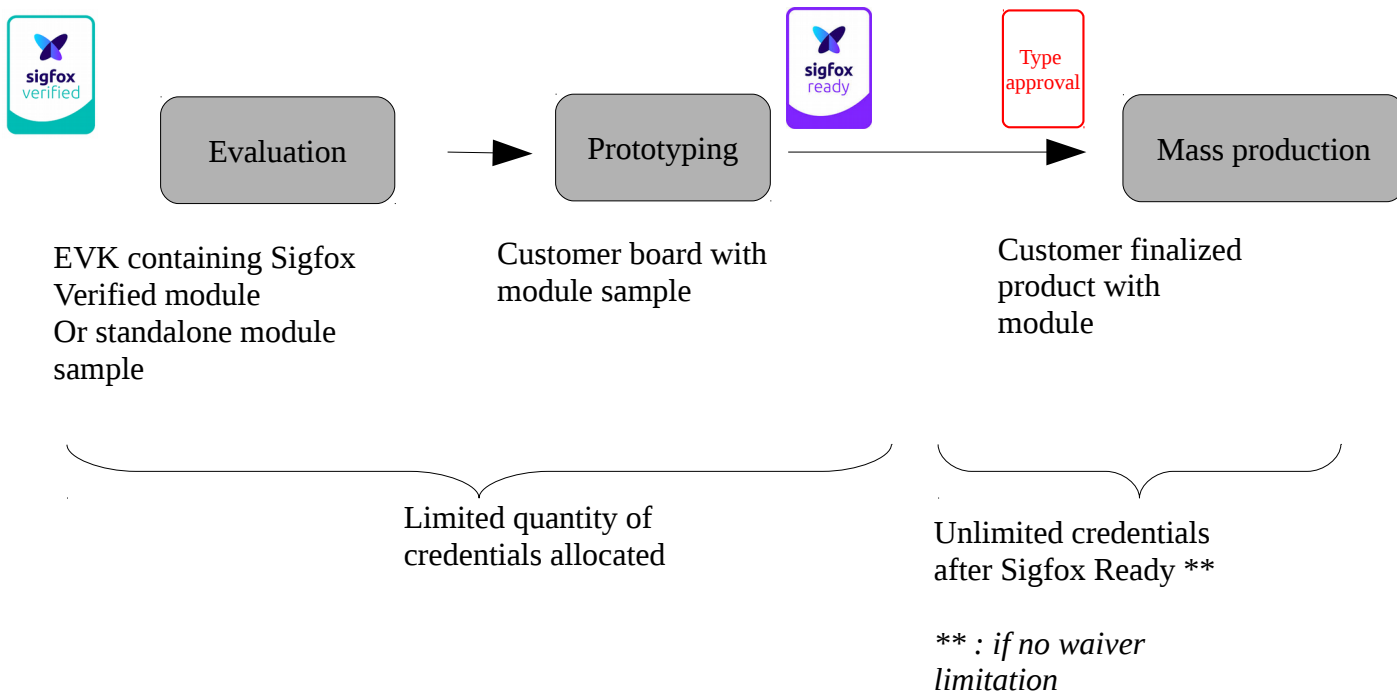
1 Document Purpose

This document explains how to obtain the Sigfox credentials provisioning (id, NAK (key), PAC) on “Open Software Architecture” modules. Open software architectures modules are modules which do not contain any firmware inside and in some cases no credentials. They are often released with software development kit and some libraries dedicated to control all hardware features and including the Sigfox part.

This document addresses different use cases you could encounter during your device development using this kind of “Open Software Architecture” modules.

This document also explain how to manage Sigfox credentials during the different development phases of your end device.


- Evaluation / prototyping /Sigfox certification
- Type approval Certification / mass production



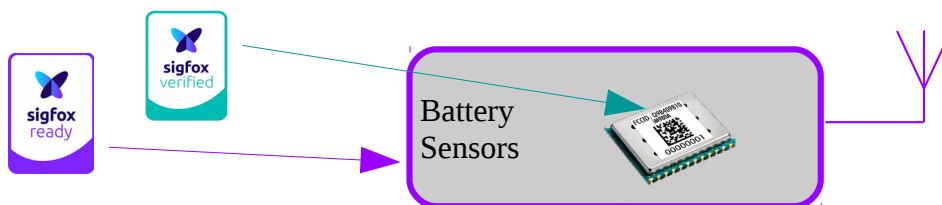
2 Conditions to access the network

You are using a “Sigfox Verified” module which has been certified by Sigfox or an evaluation kit: it means that the RF modulation/demodulation and protocol have been tested and are compliant with Sigfox requirements RF and protocol.

During the **evaluation phase**, you will have access to the Sigfox cloud with a limited number of devices registered as “prototypes” on Sigfox cloud.

 To unlock this limitation, your device must be **Sigfox Ready** certified. Therefore, to go ahead with your device based on a Verified module (power supply, application, antenna, casing), it must be submitted to Sigfox Ready[®] Certification tests to check the radiated performances.

Sigfox Ready[®] certificate is the **only way** to go to mass production and get access to unlimited amount of credentials that can be activated to be registered by your customers.



Sigfox Verified vs. Sigfox Ready



TYPE APPROVAL (FCC, RED, ARIB):

Even if there is no link between Type Approval testing and Sigfox Certification, devices must be compliant for local regulations to be massively deployed respecting spectrum access rules.

Some modules have been pre-certified for ETSI/RED and/or FCC and/or ARIB: it **does not mean that your final product is also Type-approved**, but this could help a lot to benefit of this pre-certification or modular approach.

You need to take this Type approval certification into account before going through mass production. This is not included in the Sigfox Ready / Verified certification



sigfox

Please refer to your module provider, if it has already passed through Type approval.

3 Sigfox credentials security

The most critical part of Sigfox device is the credentials. The Network Access Key (NAK) is the most critical one. It is up to device makers to store credentials securely inside the device.

Why encrypt credentials?

This operation is important to store the Sigfox credentials securely inside the module. This process allows you to manufacture devices which are unique. This operation can be fast but secret to encrypt the credentials has to be kept secured in the manufacturing process.

A slower process consists in encrypting credentials with a custom signature that will link the device to hardware.

There are many ways to secure credentials inside a device:

- **Using an external SE (Secure Element):** SDK must be compatible for SE control support.
- **Store credentials in flash and lock it in a secured memory area.**
- **Encrypt credentials with fixed global key (CREDENTIAL_KEY):** keep secret this key on device and flashing bench.
- **Encrypt credentials with unique key coming from hardware signature on each module (HW pinout, serial number, etc.)**

Why use a signature mechanism?

For security reasons, using a signature of the module to decrypt embedded credentials makes these credentials only usable on the dedicated module with a unique identifier. No cloning is possible with this mechanism.

This signature mechanism needs to run an embedded software to sign the module: this operation could take some time which is critical during manufacturing.

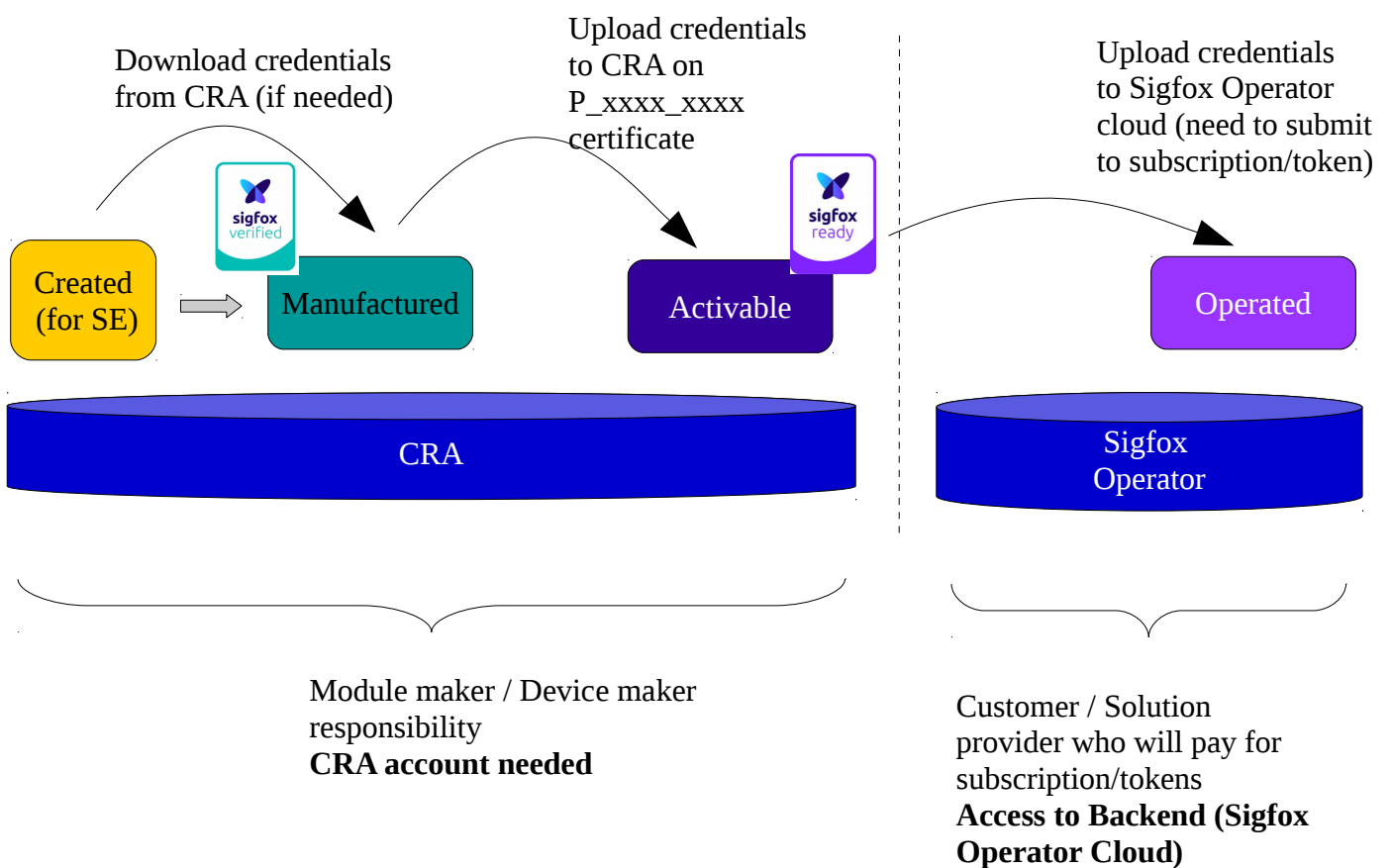


Anyway, signature is a way to link the HW with the credentials inside. This mechanism adds a constraint on the software to run only on one device: no dump or clone possible.

4 Credentials states

Sigfox credentials have different states on CRA corresponding to:

- provisioning for production (SE based or own credential provisioning)
- availability on network to be ready to be registered on cloud by customer



5 General purpose: Open architecture based module

Some modem makers have obtained a **Sigfox Verified** certificate for their hardware module with a specific software release. These modules are based on an open architecture software environment with libraries. The SDK is provided by the MCU maker or module maker and integrates all libraries to use the module on Sigfox Network. It is possible for the user to easily put its own application software inside this module and manage the Sigfox credentials inside by himself.

5.1 Evaluation Provisioning with module and EVK (Evaluation kit)

During evaluation, modules are available through EVK (Evaluation Kits) that contain Open architecture modules, or by sampling Modules to be sold on PCB. An SDK (Software Development Kit) is available for controlling these modules.

5.1.1 Open architecture module personalization

During evaluation, you can face many use-cases concerning credentials:

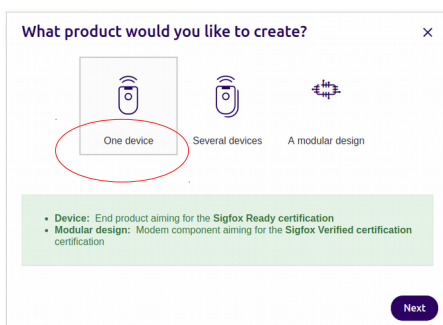
- Credentials are inside the module (using SE, using SOC (System On Chip), using pre-flashed memory)
- Credentials are **not in the module**
- Credentials are in the module **for evaluation**
- Credentials are in the module **for mass production only**
- Credentials are **never released** with modules

There are 2 ways to get credentials for an open architecture based module.

- **Module WITH credentials access:**
 - Module maker is providing **samples containing credentials** flashed inside the module (SE, SOC, pre-flashed memory). These credentials are managed by the module maker.
 - Module maker is providing a way to **personalize module** following a specific procedure and also a way to get some provisioning (web account interface to get credentials, tools etc.)

=> Nothing specific to do, just follow the module maker provisioning procedure.

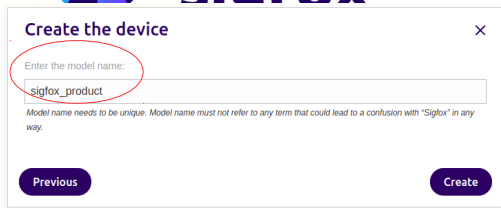
- **Module WITHOUT credentials accesses:**
 - Module maker is providing a way to **personalize module** following a specific procedure **without providing credentials accesses**.
 - Please follow this link to get credentials as a new device :
 - procedure is described here:
 - <https://build.sigfox.com/steps/development#network-credentials>
 - Connect and create a device on build.sigfox.com



Choose One device

Click on next





Enter the model name:
sigfox_product

Model name needs to be unique. Model name must not refer to any term that could lead to a confusion with "Sigfox" in any way.

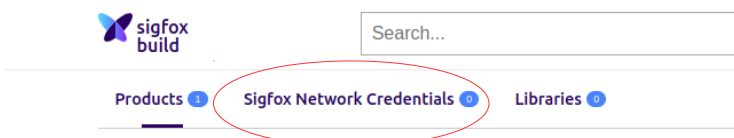
Previous Create

Enter the name of the product

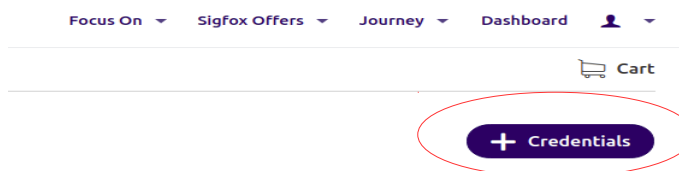
Then click on "Link to an organization"

Fill in the form and save

Then click on Sigfox Network Credentials

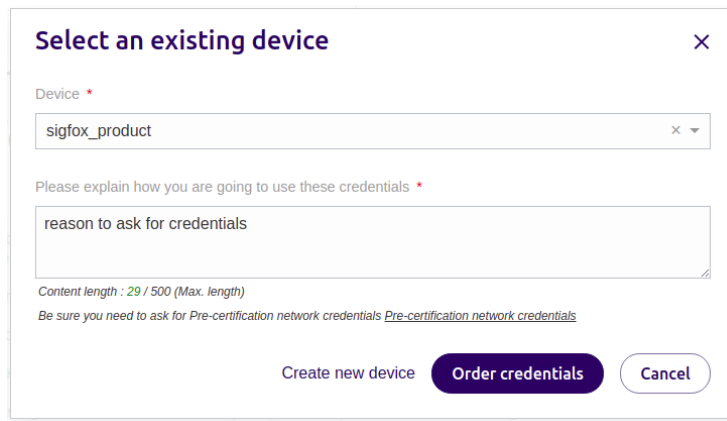


Click on "+ Credentials"



Choose the device on which you want to flash credentials

Then fill in the reason to ask for credentials and then



Select an existing device

Device *
sigfox_product

Please explain how you are going to use these credentials *
reason to ask for credentials

Content length : 29 / 500 (Max. length)
Be sure you need to ask for Pre-certification network credentials [Pre-certification network credentials](#)

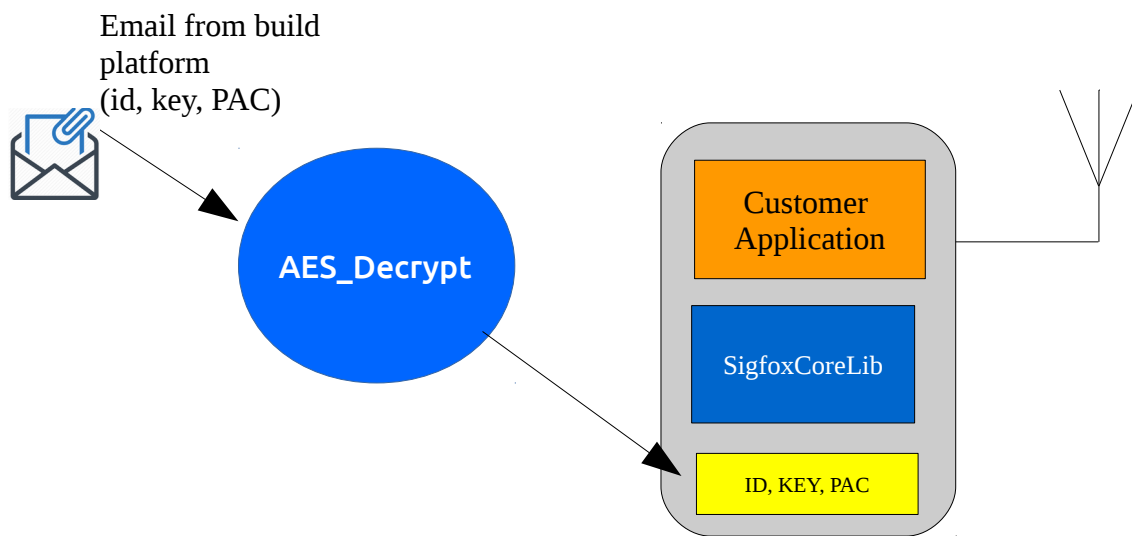
Create new device Order credentials Cancel



⇒ You will receive your credentials after validation through email.

What to do with received credentials ?

- use the AES_Decrypt tool to extract id, key, and PAC received by email from Sigfox Build.
- flash [id, key, PAC] into the device.



⇒ *Your device is ready to be activated on network, but need some tokens: see registration chapter.*

5.2 Provisioning and mass production with Open architecture module

5.2.1 Get access to your own credentials

Once the **Sigfox Ready** certificate is released by Sigfox (<https://build.sigfox.com/steps/certification#sigfox-certification-process>), you should receive a CRA access for your own credentials provisioning.

- CRA account with Modem maker profile (only if module do not contain any credentials)

This profile is dedicated to get id_key encrypted files + id_pac txt ASCII files

- CRA account with End product maker profile

This profile is dedicated to upload id_pac files to make devices **activable** on the network.

5.2.2 Embedded device firmware without encrypted credentials

No specific embedded software is used in existing SDK to encrypt credentials on open architecture modules (except ST EVK and Murata modules).

Credentials are stored in Flash or EEPROM (when available) and values are declared in pre-compiled directives.

```
#define ID FEDCBA98
#define KEY 0123456789ABCDEF0123456789ABCDEF
#define PAC xxxxxxxxxxxxxxxxx
```

Credentials are used for simplicity purpose in a clear format without credentials encryption. No decryption embedded software is needed to be used by device software and Sigfox library.



5.2.3 Embedded device firmware with encrypted credential

Available soon in the next AppNote release.

5.2.4 PC tool for decrypting credentials example

AES_Decrypt.cpp PC tool is only decrypting the credentials from CRA to extract them in clear format to be used by device maker for module personalization.

Compilation :

```
gcc AES_Decrypt.cpp -o AESd.exe
```

Usage :

AESd.exe <encryption CRA private key> <encrypted file input name> <decrypted file output name>

For source code and explanations : refer to

<https://build.sigfox.com/steps/industrialization#the-sigfox-credentials>

5.2.5 PC tool for decrypting re-encrypting credentials

A new version will be soon available to generate credential encrypted data into modules.

Waiting for that new release, module maker or SDK provider can release a way to secure your credentials and helping you to lock credentials inside the module memory.

Otherwise it is possible to implement the same way as it has been done in Murata/ST SDK but the corresponding device embedded firmware must be implemented also.

5.2.6 Mass production manufacturing steps with Open architecture modules

To get Sigfox credentials for mass production, you need:

- to use the Sigfox Verified module.
- To be Sigfox Ready: you have to submit your end device based on module to Sigfox Ready radiated tests following this link procedure:
 - <https://build.sigfox.com/steps/certification#sigfox-certification-process>

After Sigfox Ready certificate issuance, Sigfox provides you with:

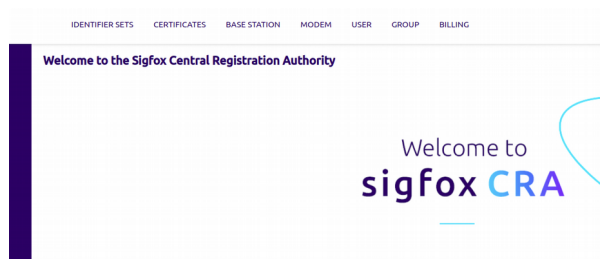
- your P_xxx certificate
- your M_xxx certificate (for credential provisioning access if needed)
- your CRA account:
 - Module maker profile
 - End product maker profile

Prerequisites:

- Have your Product certificate for Sigfox Ready.
- Have access to CRA as “modem maker”: M_xxxx_xxxx_xx
- Have access to CRA as “end product maker”: P_xxxx_xxxx_xx
- Request for a batch of credentials: *support.sigfox.com*

Manufacturing:


- Log as “Modem maker” on CRA

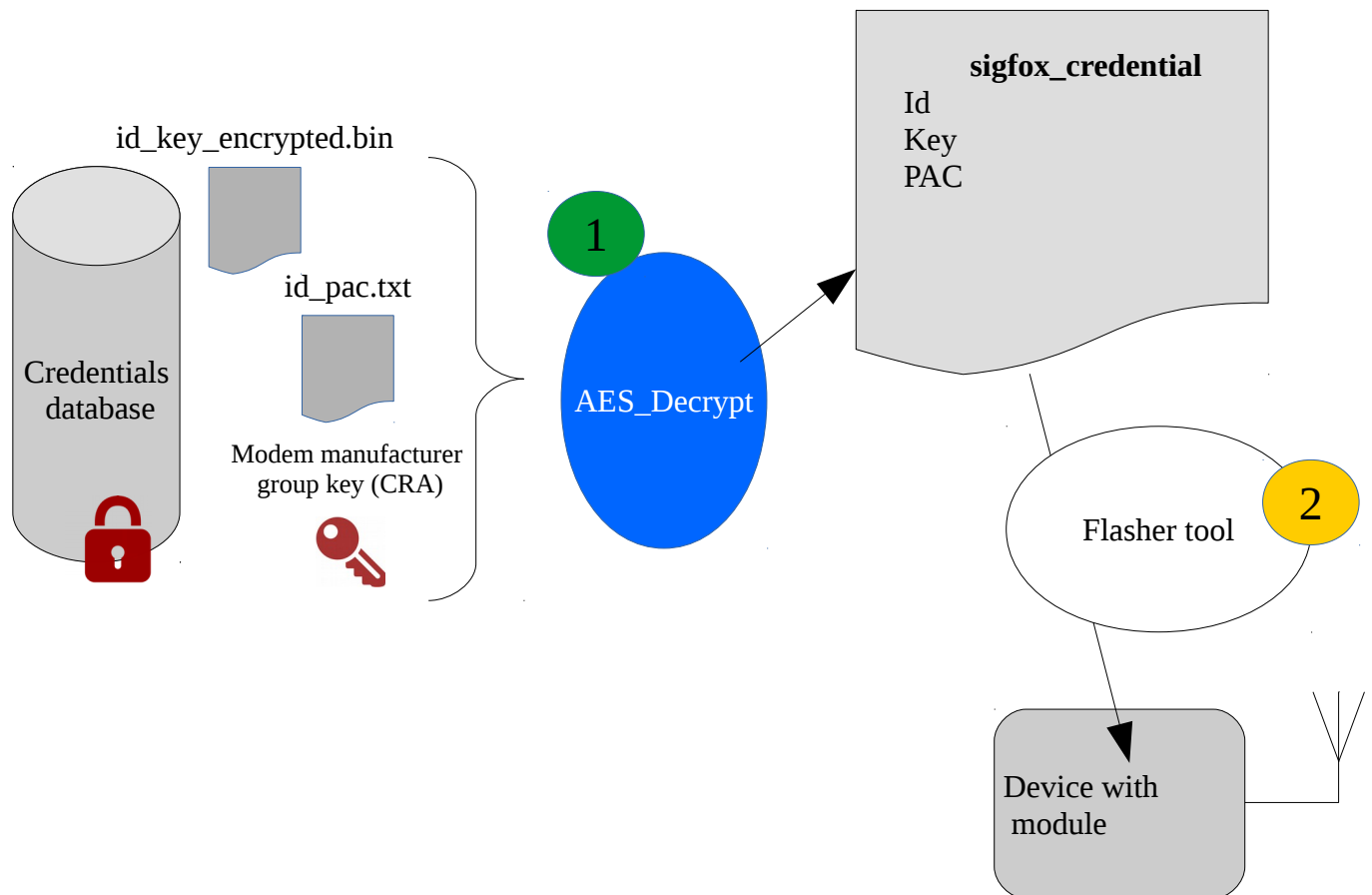


- Download the id_key.bin file

- Download the id_pac.txt file
- Build/manufacture the device
- Personalize the device

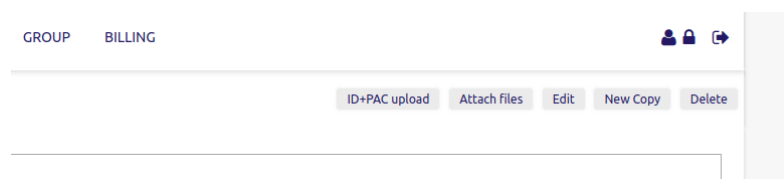
The following diagram is describing the 2 or 3 steps for manufacturing and basic setup to make a device personalization.

 It is the responsibility of the credential owner to avoid any clone manufacturing (same credentials into 2 different devices). Sigfox is not responsible for any mistake made during flashing procedure (CRC check after flashing is recommended).



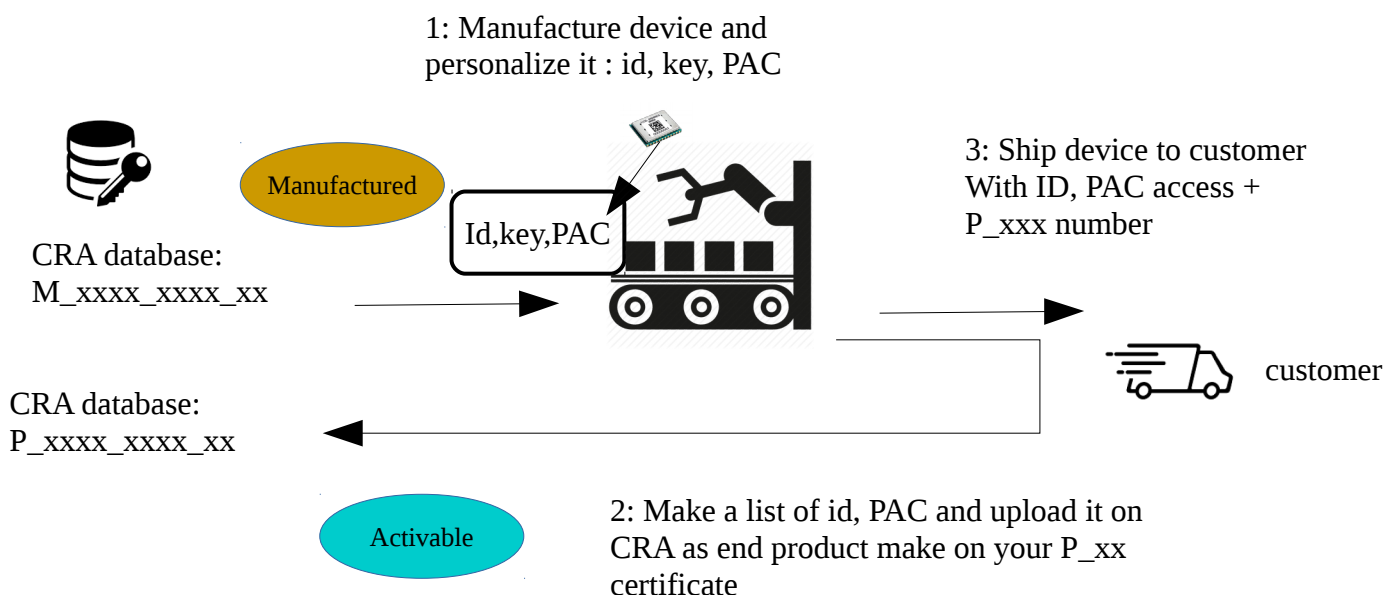
- 1 Run PC AES_Decrypt tool using the Manufacturer group key
- 2 Flash with flasher tool the generated sigfox_credential file into the right memory space

- Get ID+PAC from all devices manufactured (using SIGFOX_API_get_id, get_initial_pac or equivalent AT command) => create a list id_pac.txt
- Connect to cra.sigfox.com and log as “end product maker”.
- Open the right product certificate corresponding to the product.
- Click on ID+PAC upload



- Upload the id_pac.txt
- Products are now in “activable” state, ready to be registered on any SO by your customer.
- Ship products to your customer: you can provide the list id_pac.txt to your customer or equivalent .csv list (or give a way to get easily those info with tag reader on blister or else).

Do not forget to give the P_xxxx_xxxx_xx to your customer.





Provisioning and activation are 2 different processes. Having credentials does not mean that a token/subscription is release with them.

- Provisioning and activability of credentials required CRA accesses.
- Registration required Sigfox Operator cloud access (token/subscription)

On CRA:

To be registrable on the Sigfox network, a device (manufactured by a device maker), must be in “activable” state. This operation is **the responsibility of the device maker** as described previously.

On Sigfox Operator Cloud:

A customer is considered here as the one who will operate the device messages and pay for the connectivity (token). A device maker can also be the device messages operator (buying token from Sigfox and then customer pay for the service to this company which provides/offers an end to end solution).

6.1 Activate prototypes and EVK credentials on the network

Evaluation kits are often candidates for a temporary subscription on Sigfox network. Please refer to your module maker provider to check if subscription is included with its offer.

As for provisioning, there are 2 different ways to get some subscription during evaluation:

- Module maker describes how to activate following a specific procedure.
- Module maker invites you to use the Activate web platform using the following links:

<https://support.sigfox.com/docs/sigfox-activate:-when-and-how-to-use-it>

If this procedure do not work (your credentials are not candidate to get free prototyping connectivity), please refer to “buy connectivity” to buy connectivity token on the web platform.

6.2 Activate mass production devices on network

A device maker can have different roles:

- End device maker is only manufacturing the device HW:

This use case is only for hardware makers, that do not manage the token/subscription on Sigfox network. End device makers **never have to register devices on network!**

- End device maker is **manufacturing the device and operates** it on network

This use case is for a device maker which is providing/manufacturing devices **AND** operates them as service provider. In this use case, device maker could register but it is not mandatory at manufacturing stage. The device can be registered at installation.

Prerequisite for device registration:

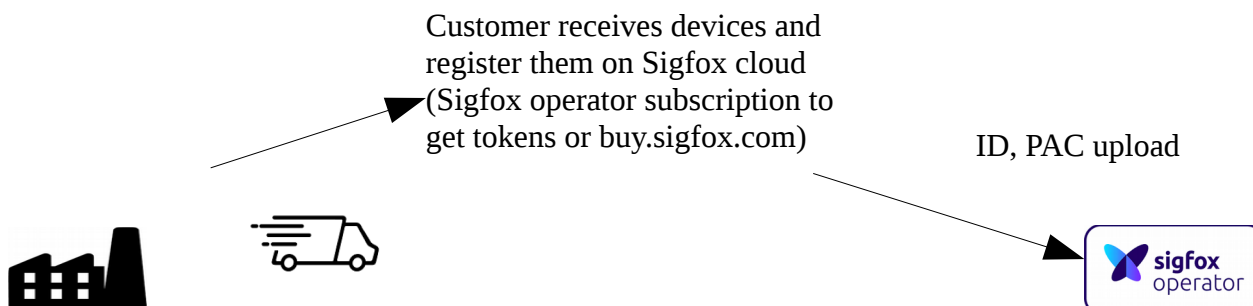
- A customer is the entity who will operate the device on the cloud and manage directly the device messages.
- Customer has to contact local SO (Sigfox Operator) to get token subscription.
- Customer can buy some token through this web site:

<https://buy.sigfox.com/>

Registration procedure:

- Customer will have to follow the link here:

<https://backend.sigfox.com/cms/section/52f8a5b593368ce020b924e1/info>





7 Annex1: Use case Murata module CMWX1ZZABZ

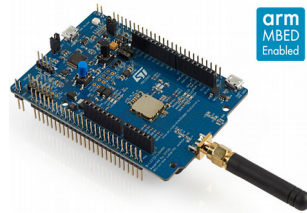
Murata LPWAN is a Sigfox Verified certified module and based on an open architecture software environment. This means that a user can easily put its own application software inside this module and manage by himself the Sigfox credentials inside.

These modules come with SDK *called X-CUBE-SFOX available at <https://www.st.com/x-cube-sfox>*

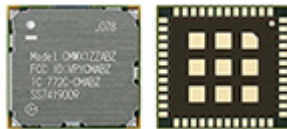
7.1 Provisioning during evaluation and prototyping with samples

Murata module is available for **sampling** through 2 ways:

- with B-072Z-LRWAN1 EVK on ST discovery board.



- Murata module CMWX1ZZABZ sampling ready to be sold on your own PCB.



In this phase, the Sigfox credentials are available through Murata web site: see provisioning with signature section.



This chapter is applicable only for Murata module sampling and EVK. This chapter is using a specific signature mechanism only valid for evaluation/prototyping.

7.1.1 Prerequisites

- Create an account on myMurata.com
- Download keil tool : <http://www2.keil.com/stmicroelectronics-stm32/mdk>
- Download X-CUBE-SFOX on st.com

7.1.2 Getting signature with EVK

- Connect the EVK with USB to a PC.
- From `/Projects/Multi/Applications/sgfx/SignatureGenerator` , Drag & Drop 'SignatureGenerator.bin' in DIS_L082Z.
- Start a terminal on COMx (win) or ttyUSBx (Linux) and copy the signature printed
- paste the printed signature on Murata web site
- sigfox_data.bin and sigfox_data.h files should be received by email

7.1.3 Personalization of Murata samples and network activation



Important note : having Sigfox credentials do not mean that subscriptions are associated. Getting credentials from Murata website will ensure to link credential to a one year free subscription.

- Build your project with the SDK integrating the Sigfox library.
- Flash the output of the previous build
- Flash the sigfox_data.bin file with STLink utility (or include the sigfox_data.h file in your project and rebuild all project)
- Get the Sigfox id and PAC: through the SIGFOX_API_get_id() and SIGFOX_API_get_initial_pac() APIs provided by Sigfox Library
- Go to <https://backend.sigfox.com/activate/>
- Click on Murata logo
- Select your country
- Enter your id and the PAC
- Enter your account detail and click on “subscribe”
- Device is ready to communicate on Sigfox network.

For more details, follow the AppNote here provided by ST:

https://www.st.com/content/ccc/resource/technical/document/user_manual/group0/d6/6e/d3/e4/72/90/49/bf/DM00406670/files/DM00406670.pdf/jcr:content/translations/en.DM00406670.pdf

7.2 Mass production provisioning: custom personalization

Once the **Sigfox Ready** certificate released by Sigfox (<https://build.sigfox.com/steps/certification#sigfox-certification-process>), you should receive a CRA access for credentials provisioning.

- CRA account as Modem maker profile

This profile is dedicated to get id_key encrypted files + id_pac txt ASCII files

- CRA account end-product maker profile

This profile is dedicated to upload id_pac files to make devices activable on the network.

In this phase, the personalization could be a long process and need to be shortened as possible. Different methods exists and are described in the following chapters in this document.

How to request for credentials on a dedicated M_xxx certificate?

Once you have the M_xxx certificate account, you should receive this information:

if you need Sigfox Network Credentials based on this certificate, you can requested them through our support portal: support.sigfox.com.

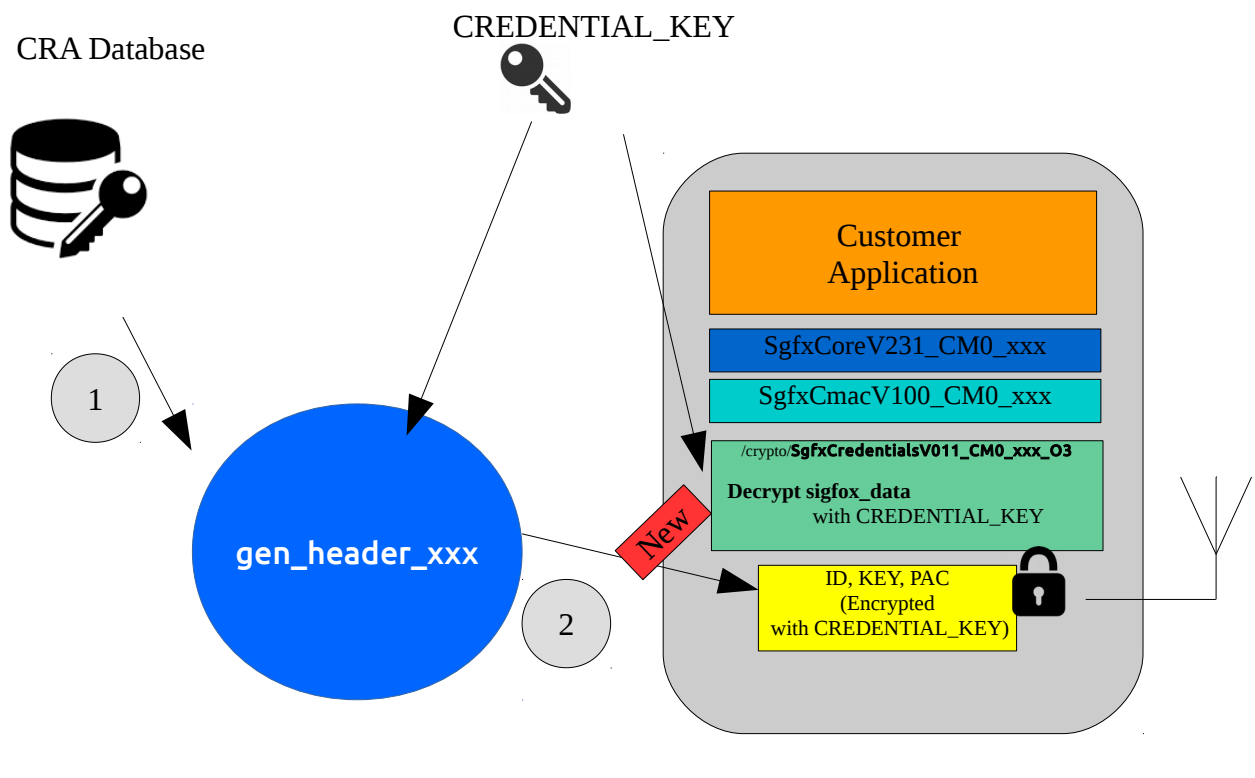
Use your login to access the service and choose Common Request, then Service Request.

Mention the following information into your request:

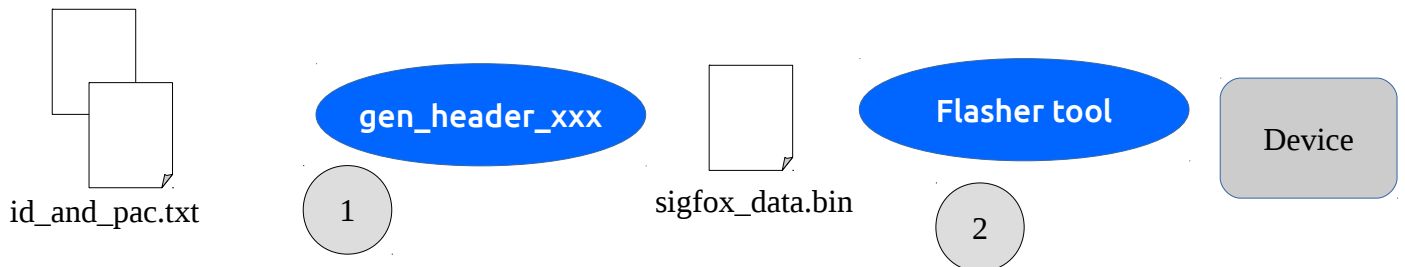
- M Certificate
- Number of IDs requested

7.2.1 Principle

Custom personalization consists in the following mechanism impacting the device firmware side and a specific PC tool.



id_and_key_encrypted.bin



7.2.2 ST SDK modifications for embedded device software

- **Uncomment line where CREDENTIAL_KEY is defined**

#define : CREDENTIAL_KEY in sgfx_credentials_template.c

in Middlewares/Third_Party/Sgfx/Crypto/ directory.

- **Personalize your CREDENTIAL_KEY (Must be the same as the one used to generate the sigfox_data.bin)**
- **From Project IDE, replace SgfxCredentialsV0yz_CM0_xxx_O3 (xxx can be GCC, IAR or Keil) by sgfx_credentials_template.c and aes_template.c**
- **Recompile Project**
- **Use the PC tool software on build.sigfox.com in annex to generate the sigfox_data.bin (flashable with ST flasher tool) or sigfox_data.h header file you can compile on SDK and reflash the complete project.**

7.2.3 Flashing / personalization bench: PC software

PC software tool is dedicated to take as input the Sigfox CRA credentials files , decrypt them and then encrypt again into a data structure to generate a sigfox_data.bin (and .h file) to be used by embedded firmware to decrypt them and use them.

Compilation :

```
gcc gen_header_murata_with_credentials.c -o gen_header
```

Usage :

```
./gen_header -h : for help
```

```
./gen_header
```

```
-k <aes cra encryption key you can find in group on CRA account>  
-x <same as CREDENTIAL_KEY in the SDK in sgfx_credentials_template.c>  
-e <0/1> 1: encrypt the sigfox_data.bin with (-x value) or do not encrypt  
-n <if you want to put a customer name: no impact on the bin generated>  
-f <cra file containing the encrypted credentials xxxxx_id_key.bin>  
-g <cra file file containing the id+PAC ASCII file containing the list of id_pac.txt  
-d <output directory name where all sigfox_data.bin and sigfox_data.h(if needed)  
will be created>
```



After running this tool, you will see that the `id_pac.txt` file is changing: this is to log and check if a device credentials have been used yet or not.

It is possible to adapt the code to make all bin in one call and then flash the list of files generated.

The attached source code for generator is managing credentials with `id_pac.txt` as database as an example. This code has to be integrated in your flashing bench system for manufacturing using a real database system to follow your production.

id_pac.txt and *id_key_encrypted.bin* files are example files to make some test before running on real batch coming from CRA.

Option:

This tool uses a hard-coded `CREDENTIAL_KEY` you will have to hide in your device firmware: this key can be static hard-coded or dynamic (linked to the hardware like signature)

Example: Try if your tool is operational with cra example files

```
./gen_header
-k 00112233445566778899AABBCCDDEEFF
-x 01010101010101010101010101010101
-e 1: encrypt the sigfox_data.bin with (-x value)
-n CUSTOM_NAME
-f id_key_encrypted.bin
-g id_pac.txt
-d directory
```

Flash the modem/device using ST link utility :

```
ST-LINK_CLI.exe -c swd ur -SE 736 -P sigfox_data.bin 0x80017000 -hardrst -v
-SE 736 Erase sector 736 (@ 0x08017000 )
```

Flashes sigfox_data.bin in the device memory



sigfox_data.bin is device specific (1 device = 1 sigfox_data.bin)

7.2.4 How to manage Sigfox credentials for manufacturing

To get sigfox credentials ,you need:

- to use the Sigfox Verified Murata module.
- To be Sigfox Ready: you have to submit your end device based on Murata module to Sigfox Ready radiated tests following this link procedure:
<https://build.sigfox.com/sigfox-ready-certification>

After Sigfox Ready certificate issuance, Sigfox provide to you:

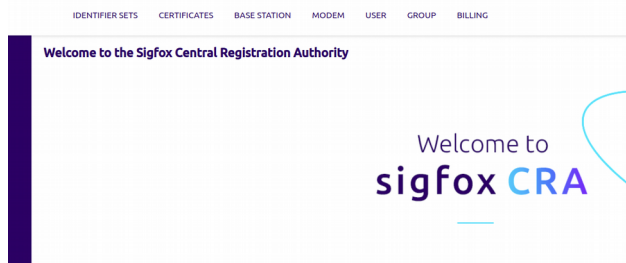
- your P_xxx certificate
- your M_xxx certificate (for credential provisioning access)
- your CRA account.
 - as modem maker profile
 - as end product maker profile

Prerequisites:

- Have your Product certificate for Sigfox Ready.
- Have access to CRA as “modem maker”: M_XXXX_XXXX_XX
- Have access to CRA as “end product maker”: P_XXXX_XXXX_XX
- Request for a batch of credentials: *support.sigfox.com*

Manufacturing:


- Log as « Modem maker » on CRA

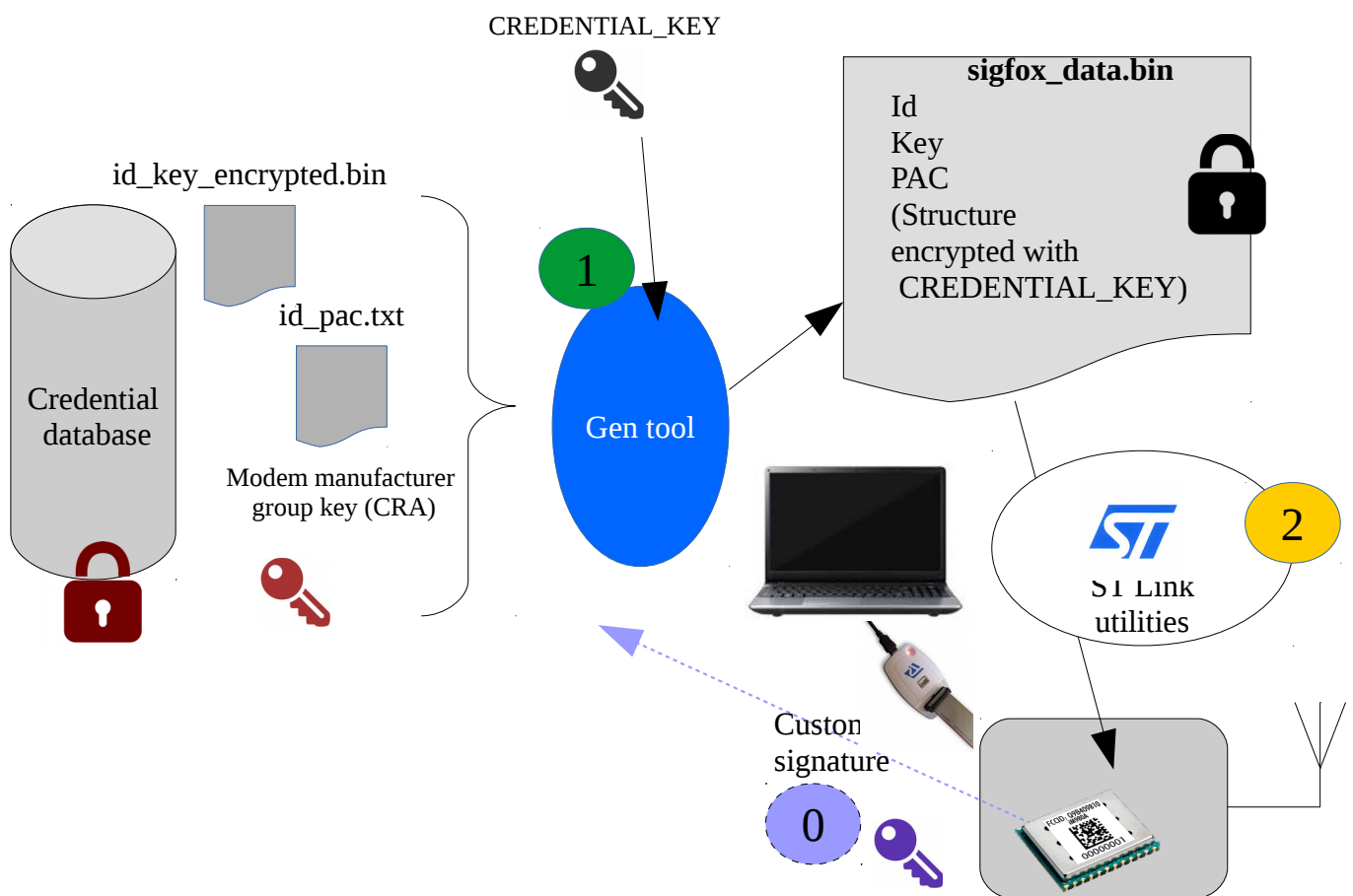





- Download the id_key.bin file
- Download the id_pac.txt file
- Build/manufacture the device

- Personalize the device

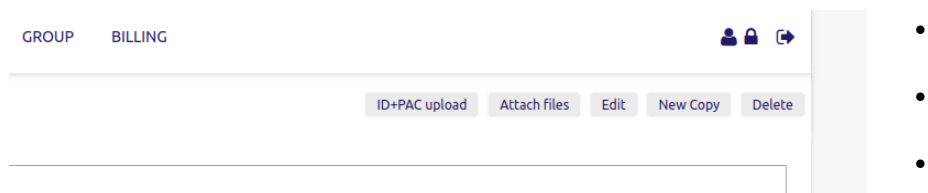
The following diagram is describing the few steps for manufacturing and basic setup to make a device personalization.

 It is the responsibility of the credential owner to avoid any clone manufacturing (same credentials into 2 different devices). Sigfox is not responsible for mistake made during flashing procedure (CRC check after flashing is recommended)



-  Option: get a signature with STLink or embed firmware on device to get signature : this signature will be used as CREDENTIAL_KEY
-  Run PC gen tool using the hardcoded CREDENTIAL_KEY or signature from device
-  Flash with STLink the generated sigfox_data.bin file into the right memory space

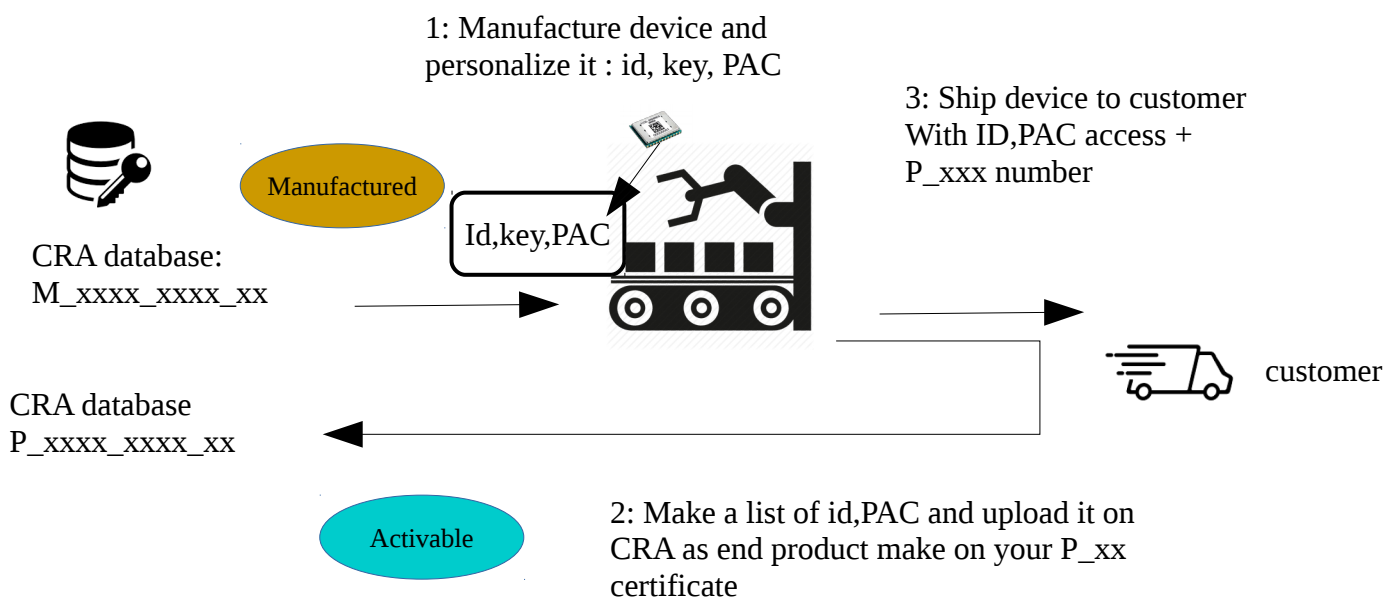
- Get ID+PAC from all devices manufactured (using SIGFOX_API_get_id, get_initial_pac or equivalent AT command) => create a list id_pac.txt
- Connect to cra.sigfox.com and log as “end product maker”
- Open the right product certificate corresponding to the product.
- Click on ID+PAC upload



- Upload the id_pac.txt
- Products are now in “activable” state, ready to be registered on any SO by your customer.
- Ship products to your customer : you can provide the list id_pac.txt to your customer or equivalent csv list.

(or give a way to get easily those info with tag reader on blister or else).

- Do not forget to give the P_xxxx_xxxx_xx to your customer.



8 Annex 2: STEVAL EVK provisioning

STEVAL EVK is not a module but is available without credentials inside. It is composed of Nucleo MCU board and S2LP RF daughter board.

ST is providing a mechanism to retrieve one credential for each EVK.

This mechanism is based on signature as described above that links the credential to the hardware it is running on.

<https://www.st.com/en/embedded-software/stsw-s2lp-sfx-dk.html>

8.1 STEVAL EVK evaluation personalization

Please follow the link here for Sigfox credentials personalization on evaluation kit based on open architecture module



https://www.st.com/content/ccc/resource/technical/document/user_manual/group0/8d/9a/ea/d7/62/06/43/ce/DM00361540/files/DM00361540.pdf/jcr:content/translations/en.DM00361540.pdf

8.2 Mass production personalization

EVK is A SIGFOX VERIFIED hardware but derived home made design ARE NOT VERIFIED.

It means:

- that any new modem hardware implementation needs to be certified
- it is not possible to go in mass production with STEVAL EVK!

Prerequisites for certification:

- <https://build.sigfox.com/steps/certification#sigfox-certification-process>
- **Submit to Sigfox RF and protocol tests to get sigfox Verified certificate**
 - <https://build.sigfox.com/steps/certification#sigfox-verified>
- **Submit to Sigfox radiated tests to get sigfox Ready certificate**
 - <https://build.sigfox.com/steps/certification#sigfox-ready>
- **(type approval certifications)**
 - <https://build.sigfox.com/steps/certification#type-approvals>
- **For provisioning, refer to this [provisioning chapter](#).**