NILE

BT 5.0 + ZB/Thread + NFC-A Standalone Module

Secure DFU over BLE application note

Version 0.1

6141 Running Springs Rd
San Jose, CA 95135
www.ivativ.com

December 22, 2019

Email: info@ivativ.com
Ph: (408) 893 7812

Table of Contents

Table of figures

1. Scope of the document

The following article shows how to securely upgrade Bluetooth application OTA (over-the-air) using signed encrypted files. The process is tested with nRF SDK v15.2. This document explains the steps in DFU, keys generation, package preparation for a DFU.

2. Overview

2.1 Introduction

The ability to update the firmware of units already deployed in the field is a common requirement for many products. For example, it may be necessary to add new features into products after the first version has been launched.

Products that use Bluetooth Low Energy technology are often designed to work with smart phones, tablets, or similar consumer electronic devices that are connected to the Internet. This makes it possible to implement OTA (Over-the-Air) firmware update capability without adding any extra cost or significant increase in the software complexity. The Internet of Things (IoT) also poses a challenge, because new types of devices and use cases are continually introduced, which can create unanticipated interoperability issues. For this reason, the ability to do OTA firmware updates is essential for any IoT device.

2.2 Security

Here we are using key, that key will be shared by the device and by the user. User need to add the key to his application which he is building. So when we do the update the device will check the key and accepts the incoming DFU request.

2.3 Reasons for firmware update

Firmware update needed for specific reasons like

For example, the Bluetooth specification is under constant development, with different revisions of the protocol stack released over time. A revision may include new features, performance optimizations, and patches to bugs or interoperability issues. It may be desirable to use the OTA update procedures to change to the latest available stack revision, even if the user application itself remains unchanged.

In other cases, the changes to be applied with OTA update are limited to the user application. Examples include fixing a security hole, or making a minor configuration change that results in improved battery lifetime. In such cases, it is desirable to update only the user application, without touching the protocol stack or other lower software layers. The Silicon Labs Bluetooth SDK supports both full and minimal updates. The difference between these two is discussed in the next section.

Note: The reader of this document should have expertise in BLE. The reader should have hands on using the Nordic nRF_SDK_v15.X.X. The readers should have knowledge to install all the required tools by following the instructions in the link attached in section 3.2

3. Requirements

3.1 Hardware requirements

- NILE DVK
- Mobile phone android or ios
- Windows PC
- Micro USB cable

3.2 Software requirements

- nRF SDK
- nrfjprog tool
- nrfutil tool
- Jlink tool
- segger embedded studio
- micro-ecc

Note: All the tools and applications must be installed in the working computer. Respective download links are attached herewith.

4. Test set up

- Here the SDK refers to the nRF5_SDK_15.2.0_XXXXXX
- Connect the DK to the PC and switch sw8 to on position

4.1 Procedure

1. Connect the DK to the PC and erase the flash using command from command line
    - nrfjprog --f nrf52 --eraseall
2. Follow the below steps to generate the keys. All the commands should give from command line.

```
# Generate a private key in c:\vault\priv.pem
nrfutil keys generate c:\vault\priv.pem

# Display the generated private key (in little-endian format)
nrfutil keys display --key sk --format hex c:\vault\priv.pem

# Display the public key that corresponds to the generated private key
# (in little-endian format)
nrfutil keys display --key pk --format hex c:\vault\priv.pem

# Display the public key that corresponds to the generated private key
# (in code format to be used with DFU)
nrfutil keys display --key pk --format code c:\vault\priv.pem
```

3. Write a public key that corresponds to the generated private key. By using the blow command
    - **nrfutil keys display --key pk --format code c:\vault\priv.pem --out_file c:\vault\dfu_public_key.c**
4. The dfu_pulic_key.c file is generated in the path of c:\vault\
5. Copy the dfu_public_key.c file from c:\vault\ to the SDK\examples\dfu\
6. Go to the path SDK\examples\dfu\secure_bootloader\pca10056_ble\ses\ and open the secure_bootloader_ble_s140_pca10056.emProject. Compile it and load it to the DK.
7. Here we have loaded only bootloader we need to load the softdevice to make an application work.
8. Load the softdevice present in the path SDK\components\softdevice\s140\hex\s140_nrf52_6.X.X_softdevice.hex by using the below command.
    - **nrfjprog --reset --program <path of soft device hex> --family NRF52 --sectoranduicrerase**
9. As we see that softdevice and secure bootloader is loaded into the DK. Now it's time to generate a package that we update via BLE.
10. Go to the path SDK\examples\ble_peripheral\ble_app_blinkey\pca10056\s140\armgcc\ from command line and compile the code by make
    - **make**
11. Make sure it is compiled without any issues
12. We need to generate package with the private key that is generated c:\vault. So copy the priv.pem key to the respective application we want to update via BLE.

13. Copy the priv.pem key to the path (in my case the path is) SDK\examples\ble_peripheral\ble_app_blinkey\pca10056\s140\armgcc\
14. Generate the package by following the command
    - **nrfutil pkg generate --hw-version 52 --sd-req <softdevice_id> --softdevice <path of the softdevice> --application-version 1 --application _build\nrf52840_xxaa.hex --key-file <path of the priv.pem key> --sd-id <softdevice_id> <name of the package>.zip**
    - For softdevice id check the link and confirm the id based on your softdevice version

Note: This command is to generate a package for application only. We can also update softdevice or softdevice + application. But there will be some changes in the command. Also make sure that this document explains only about the application update. To get more info regarding this follow this link

15. After generation of the package copy the generated zip file to the mobile.
16. Download the nRF toolbox app from play store.
17. Turn on the Bluetooth in mobile phone
18. Open the nRF Toolbox app. Click on DFU
19. Select the generated zip package file by clicking on the

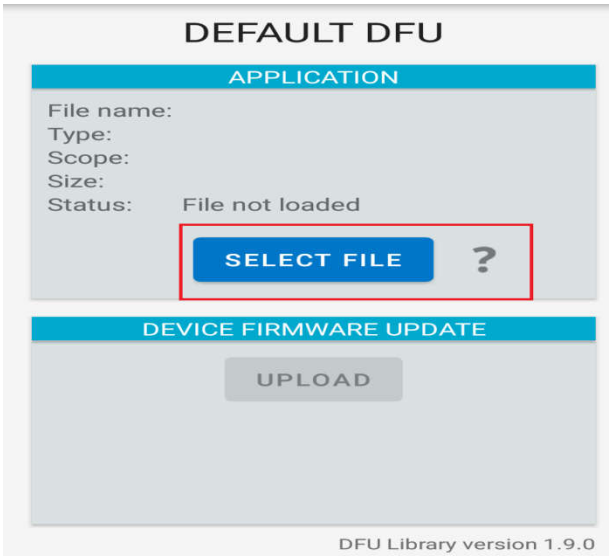

Figure 1: Indicating to select the file

20. A pop up will occur asking to select file type. Select the distribution packet and press **ok**. It will direct to the folders, Search for the package we generated.
21. Sometimes a pop-up asks to select scope, so you select **application only** and continue.
22. Now make sure our DVK is in DFU mode. **By default the DVK is in DFU mode because we have not loaded any application rather than bootloader**. If a device is in DFU you will see the LED1 and LED2 will be lit.
23. If there is any application in flash. Hold Button 4 during startup to prevent the bootloader from starting the application and force it to enter DFU mode instead. (Presently we no need this our DK is in DFU mode only it does not have any application)
24. Select the device by clicking on the icon "SELECT DEVICE" at bottom, You will see our device searching for the devices

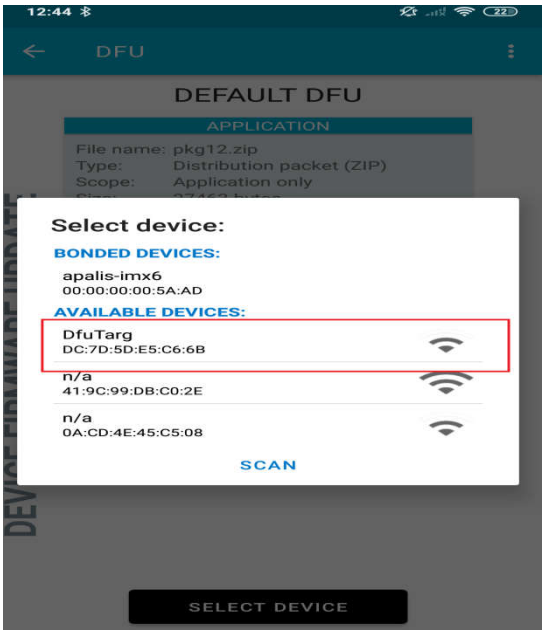

Figure 2: Showing target device

25. You will see the device with **DfuTarg**. Click on it.
26. Click on the **Upload** icon which is in middle.
27. The DFU starts you see that LED 1 and LED3 will lit when the DFU transfer is going on.

Figure 3: Showing the status of the DFU

28. The status of the DFU is also seen in mobile application.

Note: Here DFU transfer accepts if and only if the keys are matched. If any mismatch in key the DVK could not initiate the DFU transfer.

29. After the transfer is complete, the device restarts with the new application we loaded.

Note: In case you want to update a new application again, you need to repeat from step 10.

5. Test reports

Observed the DFU is happened, and the device is loaded with the new application.

6. References

- Nordicsemi - https://www.nordicsemi.com/
- Infocenter-nordicsemi - https://infocenter.nordicsemi.com/index.jsp
- Git hub - https://github.com/NordicSemiconductor/pc-nrfutil/blob/master/README.md