1. Create folder:

C:\nRF5\_SDK\_16.0.0\_98a08e2\examples\ble\_peripheral\ble\_app\_buttonless\_dfu\_spout\pca10056\s140\ses\Test\_dfu\Test\_dfuV16

I used the private and public keys that I generated before:

1. **~~Generate own private key1 in Test\_dfu1: open dos command screen and do:~~**

~~nrfutil.exe keys generate private1.key~~

1. **~~Generate public key based on your private key.~~**

~~nrfutil keys display --key pk --format code private1.key --out\_file public\_key1.c~~

~~~~

1. **~~Compile the uECC library.~~**

~~uECC library is needed for the bootloader to decrypt the signature. uECC is an external library and have to be downloaded from github here. Note: there is a license requirement comes with it.~~

~~You have to clone/download the library to the SDK folder: SDKFolder\external\micro-ecc\micro-ecc . It should look like this:~~

~~Next step, inside SDKFolder\external\micro-ecc\ choose the IDE and compiler that you want to use to build the bootloader. In my case I chose SDKFolder\external\micro-ecc\nrf52\_keil\armgcc and type make to start building uECC.~~

1. **Build the secure buttonless bootloader with dfu\_public\_key.c**

The bootloader is located at C:\nRF5\_SDK\_16.0.0\_98a08e2\examples\dfu\_spout\secure\_bootloader\pca10056\_s140\_ble\ses

#define NRF\_DFU\_BLE\_REQUIRES\_BONDS 1 // Donald

#define NRF\_SDH\_BLE\_SERVICE\_CHANGED 1 // donald

Update include **dfu\_public\_key.c** and build project.

Build and Copy C:\nRF5\_SDK\_16.0.0\_98a08e2\examples\dfu\_spout\secure\_bootloader\pca10056\_s140\_ble\ses\Output\Release\Exe\secure\_bootloader\_ble\_s140\_pca10056.hex to C:\nRF5\_SDK\_16.0.0\_98a08e2\examples\ble\_peripheral\ble\_app\_buttonless\_dfu\_spout\pca10056\s140\ses\Test\_dfu\Test\_dfuV16

1. **Copy softdevice.hex to Test\_dfuV16 folder**

Open the folder C:\nRF5\_SDK\_16.0.0\_98a08e2\components\softdevice\s140\hex, and copy the .hex file s140\_nrf52\_7.0.1\_softdevice.hex to C:\nRF5\_SDK\_16.0.0\_98a08e2\examples\ble\_peripheral\ble\_app\_buttonless\_dfu\_spout\pca10056\s140\ses\Test\_dfu\Test\_dfuV16 folder.

1. **Build application and copy to Test\_dfuV16 folder**

At folder: C:\nRF5\_SDK\_16.0.0\_98a08e2\examples\ble\_peripheral\ble\_app\_buttonless\_dfu\_spout\pca10056\s140\ses

#define NRF\_DFU\_BLE\_BUTTONLESS\_SUPPORTS\_BONDS 1 // donald

Create an application, ble\_app\_buttonless\_dfu, and copy the ble\_app\_buttonless\_dfu\_pca10056\_s140.hex file as you did for the bootloader to your C:\nRF5\_SDK\_16.0.0\_98a08e2\examples\ble\_peripheral\ble\_app\_buttonless\_dfu\_spout\pca10056\s140\ses\Test\_dfu\Test\_dfuV16

1. **Generate settings page in** C:\nRF5\_SDK\_16.0.0\_98a08e2\examples\ble\_peripheral\ble\_app\_buttonless\_dfu\_spout\pca10056\s140\ses\Test\_dfu\Test\_dfuV16 folder

**nrfutil settings generate --family NRF52840 --application** ble\_app\_buttonless\_dfu\_pca10056\_s140.hex **--application-version 1 --bootloader-version 1 --bl-settings-version 1 settings1.hex**

The output looks like this:



1. **Merge the setting1 with bootloader1**

Merge the **settings1.hex** and **secure\_bootloader\_ble\_s140\_pca10056.hex** together by running this command in **cmd** in your dfu folder:

mergehex.exe --merge settings1.hex secure\_bootloader\_ble\_s140\_pca10056.hex --output settings1\_and\_bootloader1.hex

The output looks like this:



1. **Merge the DFU packet and build app\_buttonless1.zip**

Generate the DFU packet by running this command in **cmd** in your dfu folder:

nrfutil pkg generate --hw-version 52 --application-version 1 --application ble\_app\_buttonless\_dfu\_pca10056\_s140.hex --sd-req 0xA9 --key-file private1.key app\_buttonless1.zip

The output looks like this:

****

1. **Programming bootloader**

Open the dfu folder in **cmd**, and write the commands:

nrfjprog.exe -e

nrfjprog.exe --program s140\_nrf52\_7.0.1\_softdevice.hex --verify

nrfjprog.exe --program settings1\_and\_bootloader1.hex --verify -r



1. **Start nRF Connect; NO DfuTarg**



1. **The Nordic nRF52840-DK tas LED1 and LED2 both blinking**

****