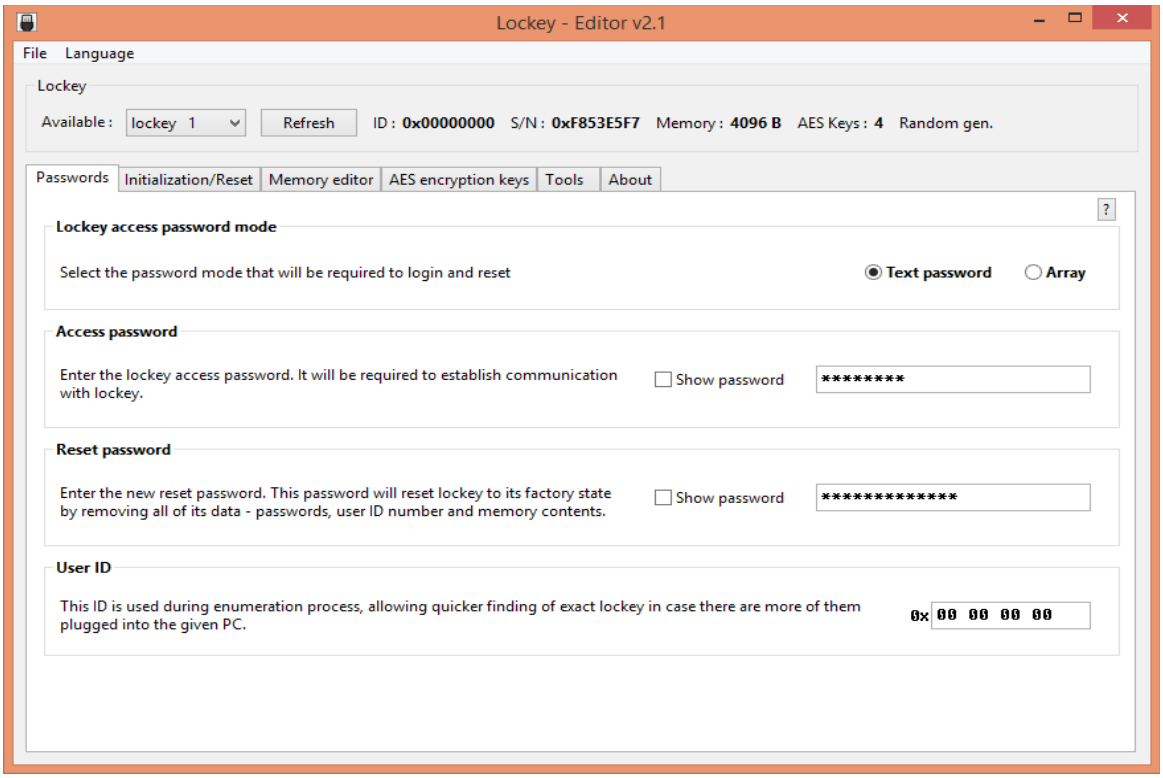
Software protection dongle

editor application

Date 07/03/2020

Contents

About editor



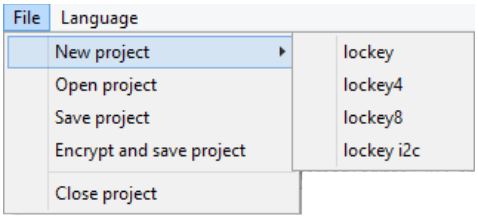
Editor is designed to simplify configuration process of **lockey**. You can use it to:

- Initialize new (empty) **lockey**
- reset **lockey** (restore it to factory settings) that has been previously initialized
- read and write user accessible memory
- store AES keys for use with built-in encryption / decryption
- test encryption / decryption with chosen AES key
- generate pseudo-random data blocks within **lockey**

With editor you can create passwords needed for initialization and save them in project file, together with other data, such as memory contents or/and AES keys. Project file can be encrypted and password protected.
Editor also can be used to mass-product **lockeys** to be used to protect particular application, for example.

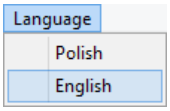
Application’s menu

Menu consists of following parts:



Project file management.

New project	Creates new project for chosen **lockey** type.
Open project	Opens existing project file. If project is encrypted, application will ask for entering password.
Save project	Saves edited data to project file. Data consists of initialization/reset passwords, memory contents, AES keys.
Encrypt and save project	Similar to previous option, but before writing file to disk encrypts its contents. Will ask for password before proceeding.



Language selection.

Working with application

After opening application you need to select one of the following from the **File** menu:

- **New project** – to create new project for chosen **lockey** type

or

- **Open project** – to open previously created and saved project (also if it was encrypted)

After selecting option application will present working project in tabs, segregating thematically available options and settings.

List of available lockey dongles

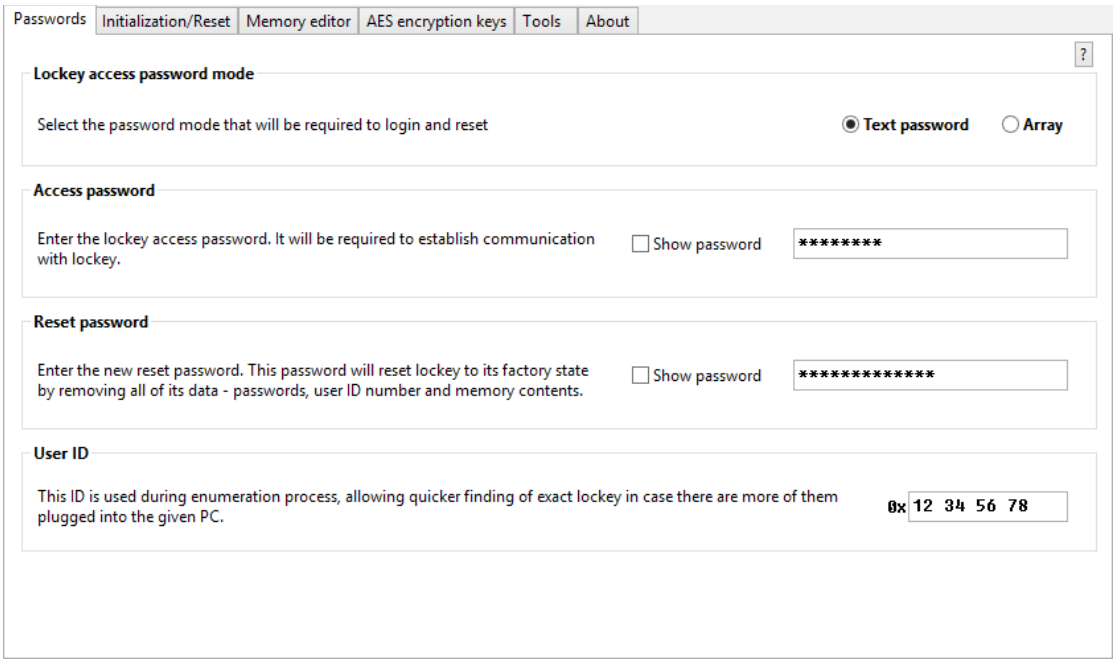
List of available **lockeys** plugged into USB ports is accessible in **Lockey** group:

Editor operates on selected **lockey**. You can change selection by opening **Available** pull-down menu and choosing other entry.

If the list is empty check if **lockey** is properly connected to computer and click **Refresh** button.

After selecting device from list application automatically reads and shows its id number, serial number, memory size, number of encryption keys possible to store and also if pseudo-random generator is available.

Passwords tab



Lockey bought from manufacturer is empty (not initialized) and thus it needs initialization before use. Initialization sets communication and reset passwords, and, optionally, id number.
In this tab you need to choose method of entering password and fill required fields.

NOTE

If you lose communication and reset passwords for **lockey** after initializing it, and cannot remember them, such **lockey** *becomes* ***useless***.

After creating new project and entering passwords, or after modifying them, it is strongly advisable to immediately save project with ***File/Save project****.*

It helps avoiding locking access to **lockey** *out.*

Methods of entering passwords

Login method (*see API programmer's reference - „lockeyapi.pdf"*) creating encrypted connection with **lockey** dongle uses communication password. This password can be passed as:

- open text

or

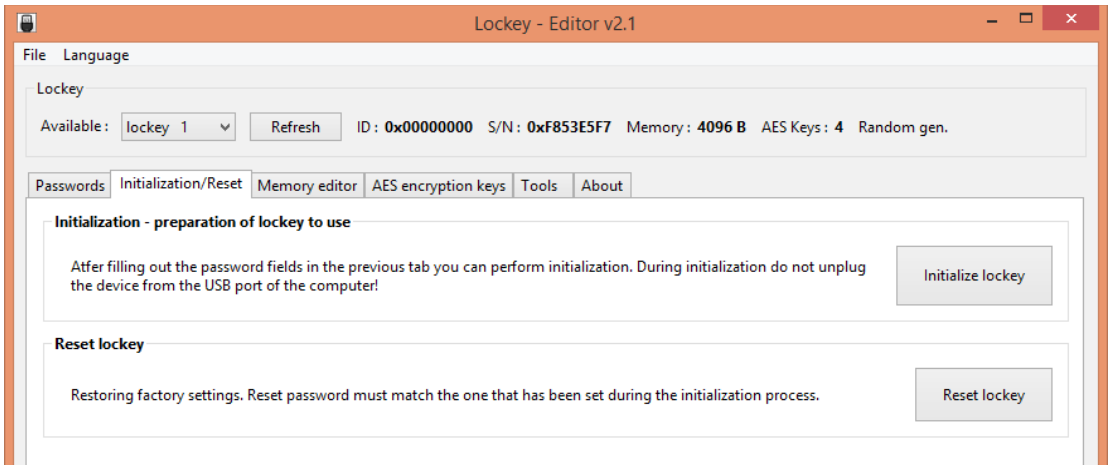
- 32-byte array

Above applies also to the **Reset** method.
When creating a new project, you must decide what type of password you will further use to establish communication with **lockey**.

Id number

If you want to use enumeration feature of picking up only **lockey** devices with particular id numbers, you need to provide one here (*see information about **ClockeyEnum** class in programmer's reference - „lockeyapi.pdf"*).

Initialization/Reset



The buttons:

Initialize lockey

Sets in **lockey** values from previous tab:

- communication password (key)
- reset password (key) – for restoring factory settings
- optionally – **lockey** id number, or 0 if not set.

Initialization could be done only if one of the following applies:

- **lockey** has not been initialized before, i.e. is in state as from manufacturer
- restoring factory settings with **Reset** has been performed succesfully

Reset lockey

Restores factory settings of **lockey**, erasing:

- communication and reset passwords
- data from memory
- encryption keys
- id number

Requires correct reset password, that has been set during initialization of this **lockey**.

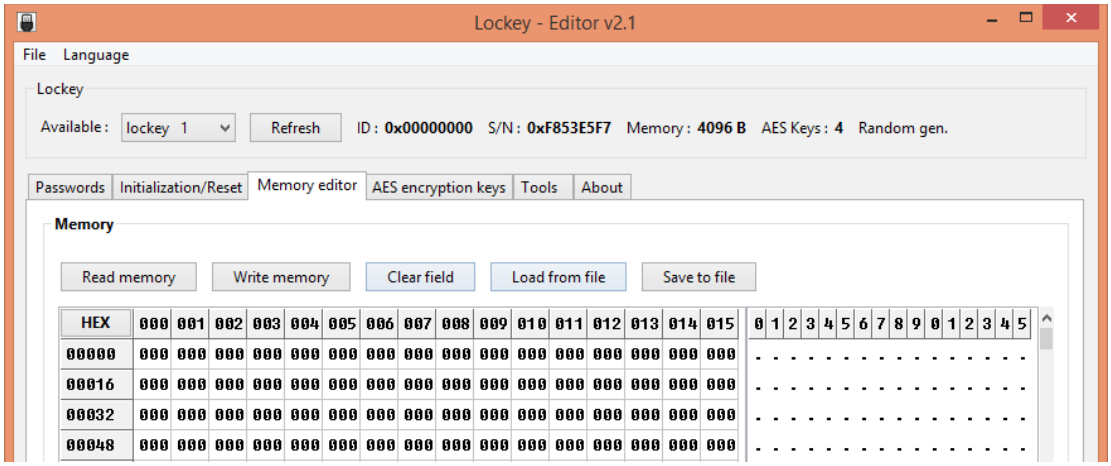
NOTE

If you lose communication and reset passwords for **lockey** after initializing it, and cannot remember them, it means that:

- *there is no way to access such **lockey***
- *there is no possibility to restore factory settings of such **lockey***

Due to the length of password arrays and strength of used AES algorithm, there are no known methods of cracking passwords in a reasonable time.

Memory editor



This tab contains functions for reading, writing and modifying memory contents of **lockey**.

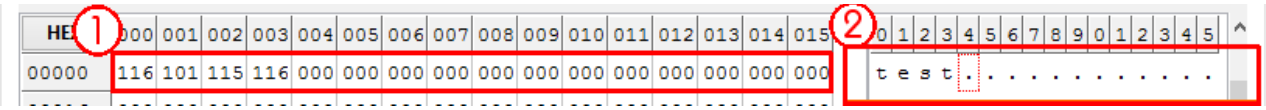
Each single memory cell can hold one byte of data. Each byte of the memory has its address. For convenience editor splits all memory into 16-byte blocks.

Memory editor functions:

Read memory	Reads user accessible memory contents of **lockey**, that has been selected from **Available** pull-down menu.
Write memory	Writes all memory contents to selected **lockey**.
Clear buffer	Zeroes all of memory in application's buffer.
Open file	Opens selected binary file and reads its values to application's buffer. If file is longer than buffer size, only part of the file is loaded.
Save file	Saves memory contents as binary file.
HEX/DEC	Toggles mode of displaying numbers – hexadecimal/decimal

Editing memory contents

Each of the memory cells can be modified by entering at chosen address new value, by number (1) or by corresponding ASCII character (2):



After finishing editing memory in buffer you can save it to **lockey** (**Write memory** button)**.**

NOTE

*Read and write operations are possible only on initialized **lockey**.*

Example of using memory

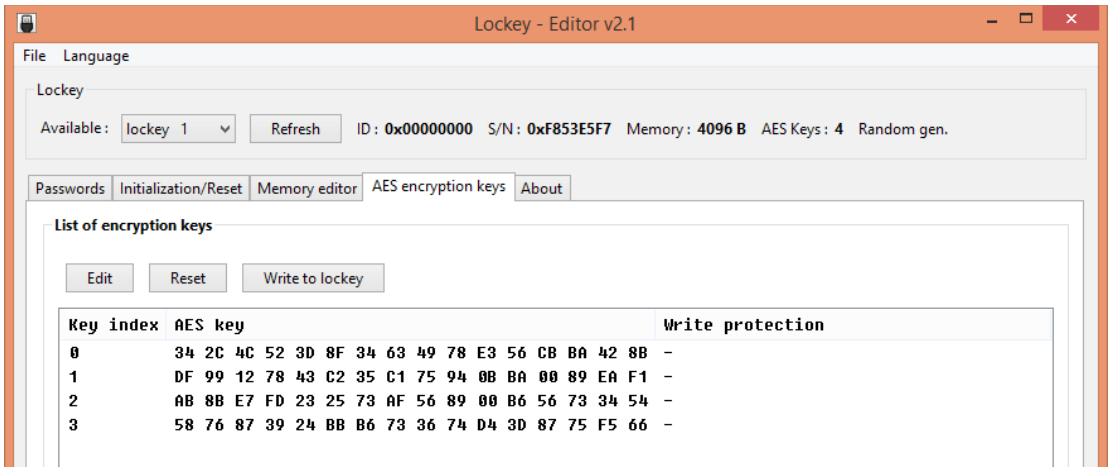
Built-in accessible memory of **lockey** can be used in protected application reading its fragments and analyze data. This way you can create flexible protection mechanisms, for example:

- licensing the number the application can run
- specifying expiration time, after which application no longer works
- creating in-application modules that has their functioning allowance tied to specified bits set in memory, limiting printing, saving files, accessing other resources, etc.

First, however, you should think about what information should be included in memory and define its layout for your particular needs.

For reading memory from code within protected application use **ReadByte** or/and **ReadMemo** methods. (*see description of **ReadByte/ReadMemo** methods in API and programmer's reference - „lockeyapi.pdf"*)

AES encryption keys



This tab contains functions for creating, editing AES encryption keys and storing them in **lockey**. Due to security reasons **lockey** does not have mechanisms of reading back stored encryption keys. Once stored, they can only be used to encrypting/decrypting data within **lockey**.

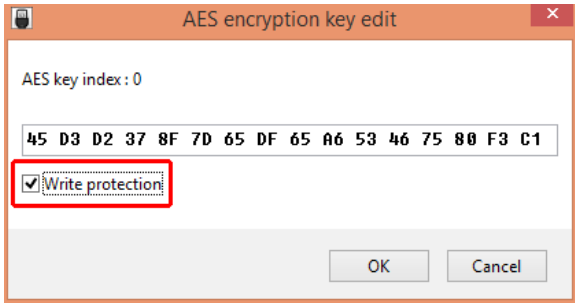
Your **lockey** needs to support cryptography features to use functionality from this tab (for example **lockey4**, **lockey8**).

Available functions:

Editj	Opens window for editing encryption key, selected on key's list.
	NOTE *Does not edit key directly in **lockey**.*
Clear	Clears (zeroes) selected encryption key.
	NOTE *Does not clear key directly in **lockey**.*
Store in lockey	Stores encryption keys in **lockey**.

Editing encryption key

Select key on list and click **Edit** button.
If **Write protection** option was additionally checked, after writing this key there will be no possibility to re-write it, erase it, or change it in any other way.



NOTE

Write protection of encryption key can only be cleared by restoring factory settings of **lockey** *and thus erasing its all contents.*

Storing encryption keys

All of the keys will be stored in **lockey** after clicking **Store in lockey** button, according to its index from the list.
If there were already stored in **lockey** encryption keys with write protection enabled, they can not be overwritten.

Encryption keys example of use

Using encryption keys from code of protected application is done using **Encrypt** and/or **Decrypt** methods. These methods as arguments need only index of stored encryption key, and data block to encrypt/decrypt (*see description of* ***Encrypt/Decrypt*** *methods in API and programmer's reference - „lockeyapi.pdf"*).

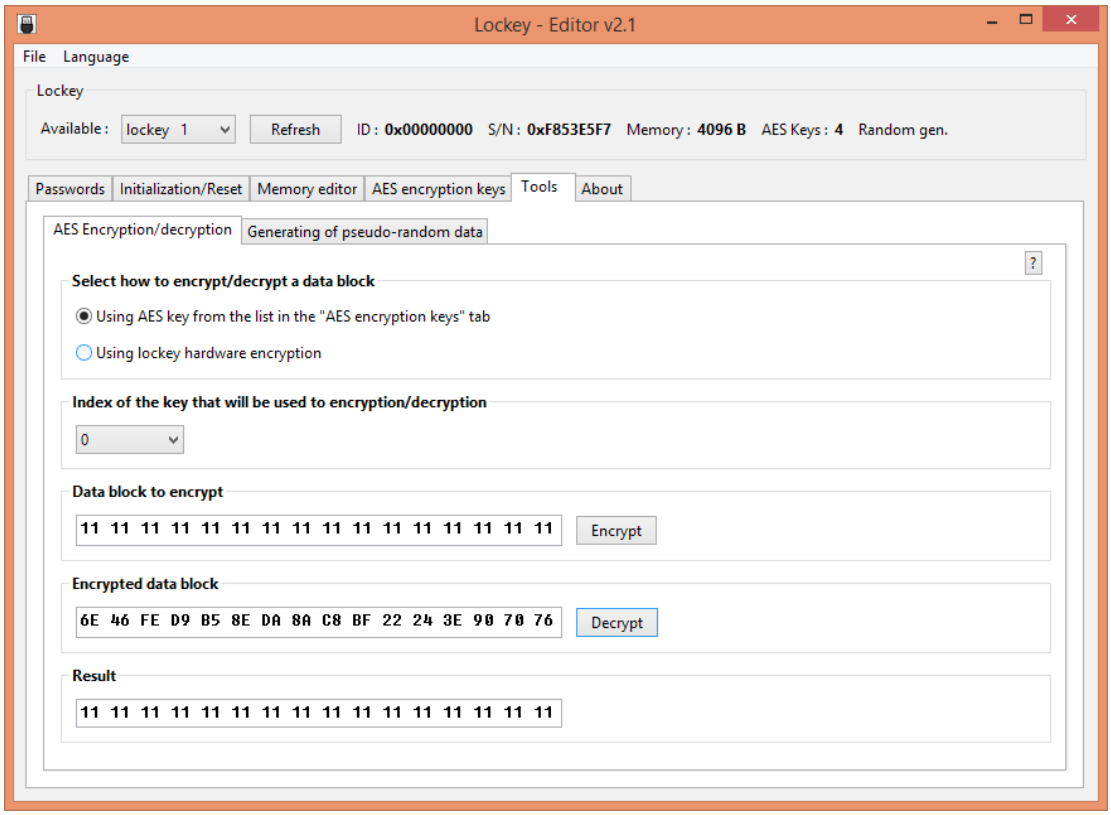
Encryption can be used:

- as part of application's protection from unlicensed use
- to protect data sent over the network
- generating electronic signatures for documents, files

Tools

This tab contains helper tools, you can use when planning/testing your application protection.

AES encryption/decryption



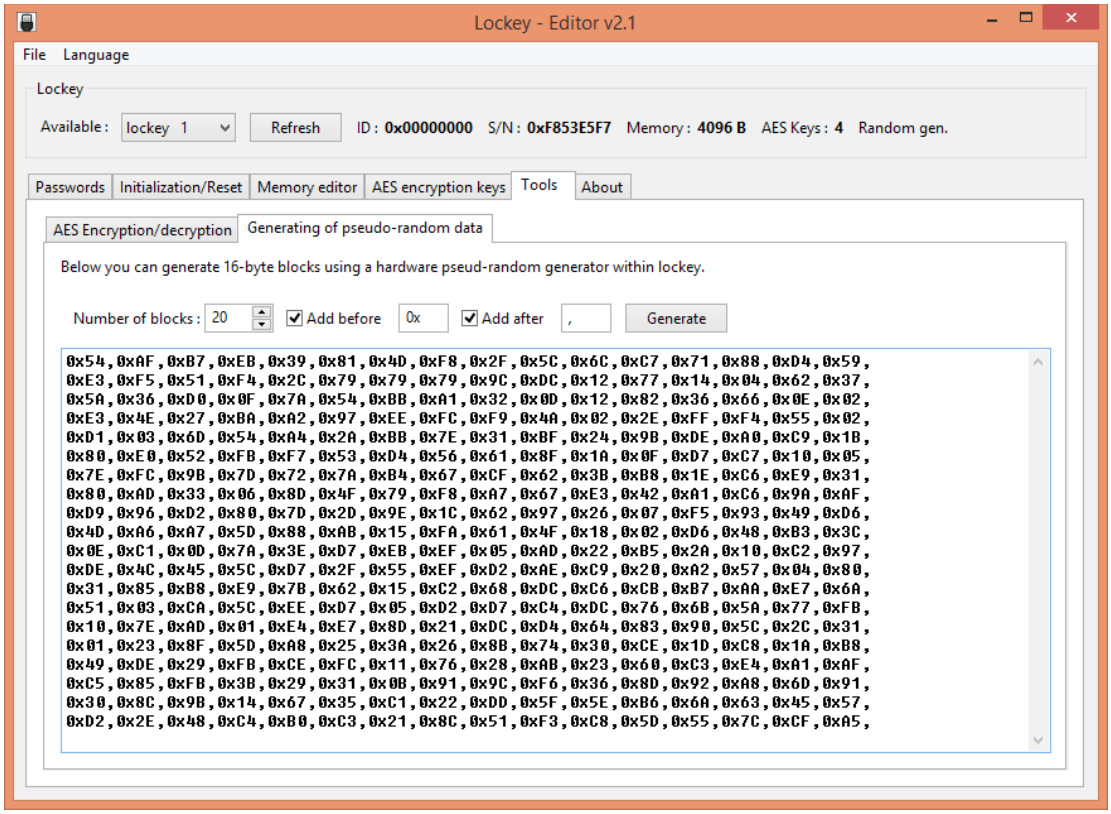
Set of tools for encrypting/decrypting single block of data using AES encryption algorithm.

Encryption can be done by application itself (software), or using connected **lockey's** built-in encryption (hardware). If former has been chosen, selected index of key corresponds to particular key in **AES encryption keys** tab. Otherwise it uses AES key at given index in **lockey**.

Example use:

- protection of sensitive data within protected application
- testing AES algorithm

Generating pseudo-random data



Using **lockey** it is possible to generate pseudo-random blocks of 16-bytes data.
They can be used to, for example, creating arrays for hiding sensitive data, or generating strong passwords, etc.
It can also be used to generate AES encryption keys.

Preparing lockey for demo projects

To learn the **lockey** API using included demo projects we recommend initializing with passwords provided with demo project. At any moment it will be then possible to restore factory settings for the **lockey** and reinitialize it with any, secret passwords.

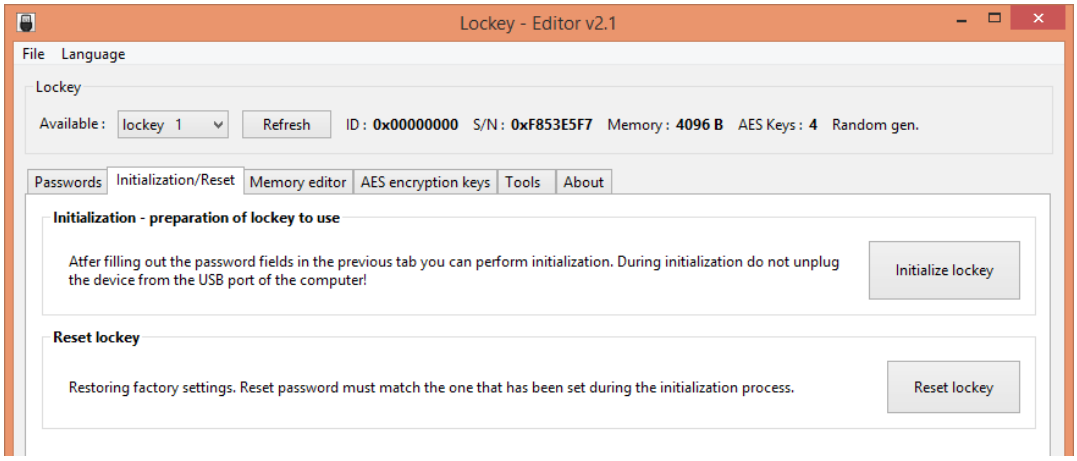
To do this, follow the steps:

1) Plug only one, not initialized **lockey** into computer's USB port.

2) Run **lockeyedit** application.

3) From the application's menu select **File/Open project** and load demo project from included file **demo.lkp**.
   On the drop-down list in the **Lockey/Available** window part there will be selected plugged in **lockey**.

4) Select **Initialization/Reset** tab



5) Click **Initialize lockey** button and wait for initialization to finish.

Reset procedure (restore factory settings) for already initialized **lockey** is similar, with the exception of the last step, instead of **Initialize lockey** button you need to click **Reset lockey**.

NOTE

*If you lose communication and reset passwords for **lockey** after initializing it, and cannot remember them, it means that:*

*- there is no way to access such **lockey***

*- there is no possibility to restore factory settings of such **lockey***

Due to the length of password arrays and strength of used AES algorithm, there are no known methods of cracking passwords in a reasonable time.

*Do not disconnect **lockey** from computer's USB port during initialization!*

Default demo project passwords:

Communication password (login): lalamido
Reset password : lalamidoreset

Notes

Notes

www: lockey.eu

e-mail: office@lockey.eu