SEkey™ GUI

Project Documentation

Release: February 2021

Proprietary Notice

The present document offers information subject to the terms and conditions described hereinafter. The authors reserve the possibility to change the content and information described in this document and to update such information at any time, without notice. Despite the attention that has been taken in preparing this document, typographical errors or omissions may have occurred.

Authors

Vahid EFTEKHARI *(PhD candidate, Politecnico di Torino)* vahid.eftekhari@polito.it
Matteo FORNERO *(Researcher, CINI Cybersecurity National Lab)* matteo.fornero@consorzio-cini.it
Nicoló MAUNERO *(PhD candidate, Politecnico di Torino)* nicolo.maunero@polito.it
Paolo PRINETTO *(Director, CINI Cybersecurity National Lab)* paolo.prinetto@polito.it
Gianluca ROASCIO *(PhD candidate, Politecnico di Torino)* gianluca.roascio@polito.it
Antonio VARRIALE *(Managing Director, Blu5 Labs Ltd)* av@blu5labs.eu

Trademarks

Disclaimer

Contents

1 Introduction

This project consists in a simple graphical user interface for the **SE*key*™** Key Management System. Before reading this manual, it is strongly suggested to read the dedicated **SE*key*™** documentation that can be found in the general **SE*cube*™** SDK Wiki[1] and in the specific **SE*key*™** Wiki[2]. A copy of the entire **SE*key*™** documentation is included with the source code of the **SE*key*™** GUI.

The GUI is composed of two components: a frontend and a backend. The frontend is the GUI built with the Qt framework, the backend is an application without any interface that runs a server that is used to gather the requests coming from the GUI for the **SE*cube*™** device.
The GUI and the backend communicate by means of a TCP socket on the port 1235; the communication through the socket is implemented by means of packets serialized by the Cereal library.
The backend is a sort of interface, for the GUI, towards the capabilities of the **SE*cube*™** device. This was implemented so that the development of the GUI was not affected by the complexity of the **SE*cube*™** SDK.
Because of this choice during the implementation, you will be required to compile two separate executables using two different IDEs.

2 Requirements

The **SE*key*™** GUI is compatible only with Linux operating systems. This is the complete list of requirements:

- Linux operating system (tested on Ubuntu 18.04 LTS, kernel 5.4.0-65-generic);
- Qt framework (tested on Qt 5.12.3);
- Qt Creator (tested version 4.14.0);
- Eclipse IDE (tested on Eclipse 12-2020)
- GCC and G++ (tested version 7.5.0)

3 Project setup

These are the steps that you need to follow in order to setup the **SE*key*™** GUI project:

1. download the **SE*key*™** GUI source code from the official website[3];
2. extract the downloaded zip file where you prefer, all paths in the next steps start from the folder where you extract the source code;
3. open Eclipse IDE and set the workspace to `/SEkey_GUI_Project/`;
4. compile the `SEkey_backend` project in `Release` mode (in case of error, right click on the project and check the properties regarding the builder and the toolchain);
5. close Eclipse and open Qt Creator;
6. open the project `SEkey_GUI.pro` in `/SEkey_GUI_Project/SEkey_GUI/`;

[1] https://www.secube.eu/resources/open-sources-sdk/
[2] https://www.secube.eu/resources/open-sources-sdk/
[3] https://www.secube.eu/resources/open-source-projects/

7. configure the project so that it is compiled at the path `SEkey_GUI_Project\SEkey_GUI\Release_Build\`;
8. compile and run the project.

Additional tips:

- the Eclipse project must be compiled with flags `-DSQLITE_TEMP_STORE=3 -std=c++17` concerning G++ and `-DSQLITE_TEMP_STORE=3` concerning GCC;
- if you encounter strange errors when compiling the Eclipse project, check the project properties and disable the build logging feature (if it is enabled);
- the Qt project must be compiled supporting at least C++14.

4 GUI features

You can run the GUI directly from the Qt Creator IDE or from the terminal of your operating system. As shown in Figure 1, the main window is very simple since it only shows the possibility to login on the **SE*cube*™** and few other options. The usage of the GUI is explained in detail in the next paragraphs.

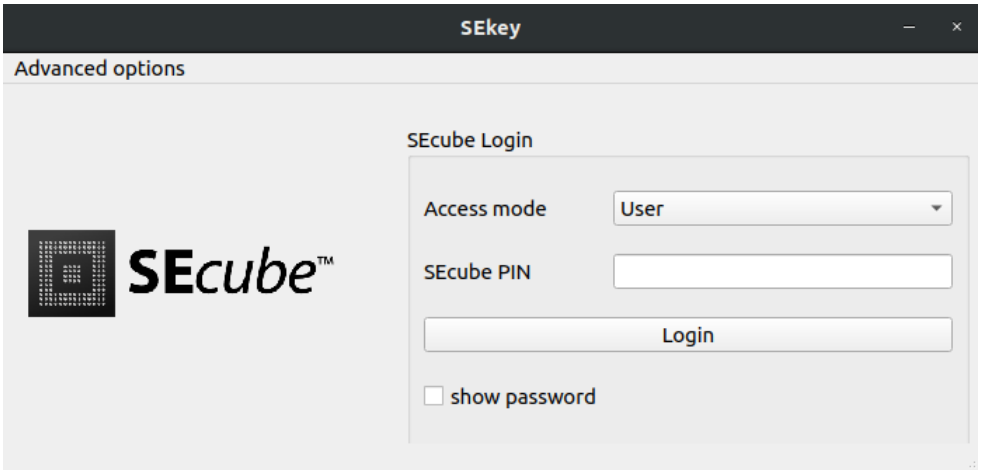

Figure 1: The login window of the program.

4.1 Advanced options menu

The advanced options menu, as shown in Figure 2, has two possibilities:

- initialize administrator **SE*cube*™** : this option must be used only once, when the **SE*key*™** administrator starts the application for the very first time. In this case, the **SE*cube*™** device of the administrator needs to be initialized with a specific procedure. The **SE*cube*™** is supposed to be brand new (meaning that you have just performed a full erase and reprogrammed the firmware). Notice that the initialization of the any **SE*cube*™** device can be done only once, if you need to initialize a **SE*cube*™** that was already initialized you must erase its flash memory and reprogram the firmware;
- set **SE*key*™** update folder: this option must be used to set the path of the folder where the encrypted files for the communication between the administrator and the users must be stored. This path, as stated in the **SE*key*™** Wiki, should be configured manually modifying a

global variable in the source code of **SE*key*™** . Since we have a GUI, it is much easier to use this option, everything you have to do is to click on that button and select the folder that is going to be shared among the administrator and the users. This should be done only once and, in general, every time that the folder is changed. This setting is non-volatile, meaning that **SE*key*™** stores the path of the selected folder in an encrypted file so that you do not need to specify it every time you launch the program.

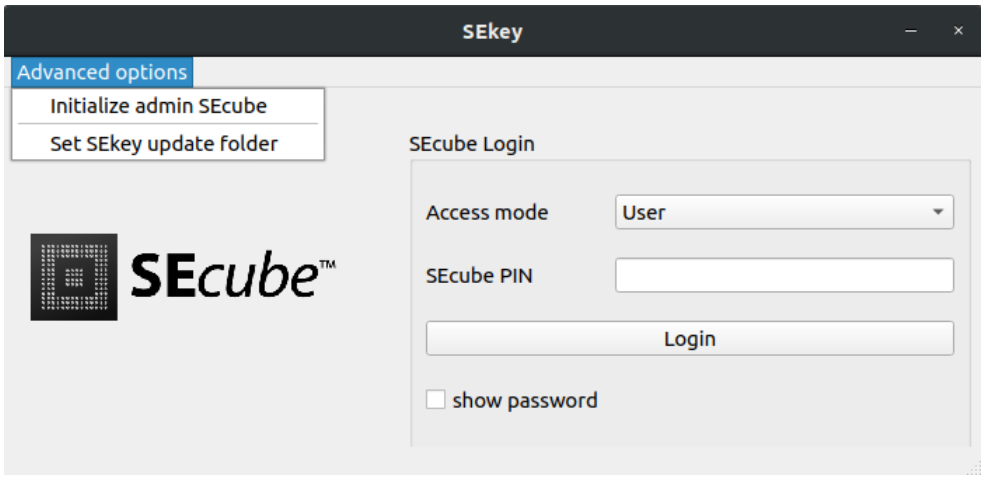

Figure 2: The advanced features menu with the option to initialize the administrator **SE*cube*™** .

4.2 Login

In order to login to the **SE*cube*™** device, you must connect the **SE*cube*™** to the PC, then you must specify the access mode and the corresponding PIN code.
Notice that, since the architecture of **SE*key*™** provides that there is only one administrator of the entire Key Management System, only the administrator actually needs to login with 'admin' privilege; everyone else must login with 'user' privilege. Moreover, the Wiki of **SE*key*™** and the Wiki of the **SE*cube*™** SDK strongly suggest that the administrator must not disclose the PIN for 'admin' access to any user of **SE*key*™** (which implies that a user of **SE*key*™** only knows his 'user' PIN).

4.3 Supported SE*key*™ features

When you successfully login as user or administrator, the most important window of the GUI opens. This window is divided into three parts:

- upper part: in the upper section of the window you have the possibility to logout, to quit, and to see details about the currently active login session (as shown in Figure 3);
- right side: on the right of the window there is a very basic menu that allows you to select which action you want to execute; these action are different for the administrator and for the users (as shown in Figure 4);
- left side: on the left of the window, as shown in Figure 5, there is a tree-view that is populated whenever you execute a feature of the GUI that implies printing some information on the screen (i.e., retrieve information about a user).

This is all you really need to know in order to use the GUI, it is very simple.

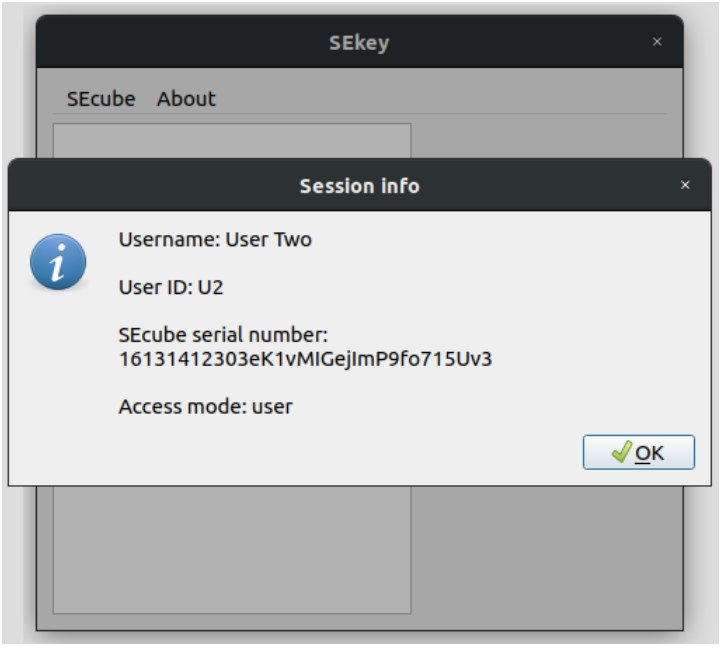

Figure 3: The window showing information about the currently active login session.

5 FAQ

- **The GUI does not start properly**: the thread used by the GUI to communicate with the **SE*cube*™** may still be running because the previous instance of the GUI was not properly terminated. Check the list of the active processes, you should see a process named 'SEkey_backend' that is listening on the port 1235. Kill the process manually before starting the GUI.
- **The GUI crashes or does not work properly after initializing the SE*cube*™ of a user**: after initializing the **SE*cube*™** of a user, the administrator must remove the **SE*cube*™** of the user when the GUI asks for removal. Remember that you must not unmount the **SE*cube*™** of the user because unmounting can corrupt the existing communication between the GUI and the **SE*cube*™** of the administrator. During our tests with Ubuntu 18.04 LTS (Virtualbox VM on Windows host) we noticed that the operating system has troubles distinguishing two **SE*cube*™** devices connected to the same PC. For some reason, unmounting one **SE*cube*™** triggers the unmounting of both **SE*cube*™** devices. This is the reason of the problems in the GUI, so do not unmount the **SE*cube*™** of the user before removing it.
- **There are files with strange hexadecimal names on the MicroSD of the SE*cube*™** : those are encrypted files that are required by the internal mechanisms of the **SE*cube*™** SDK. Do not delete them, move them or edit them.
- **The GUI is very slow**: the GUI needs to wait for the results coming from the **SE*cube*™** . Since most of these results are stored in encrypted databases, there is a significant overhead involving data decryption; moreover, performance is not the target of this open source project.
- **SE*key*™ does not work properly, there are always errors**: try to setup again the path of the update folder.

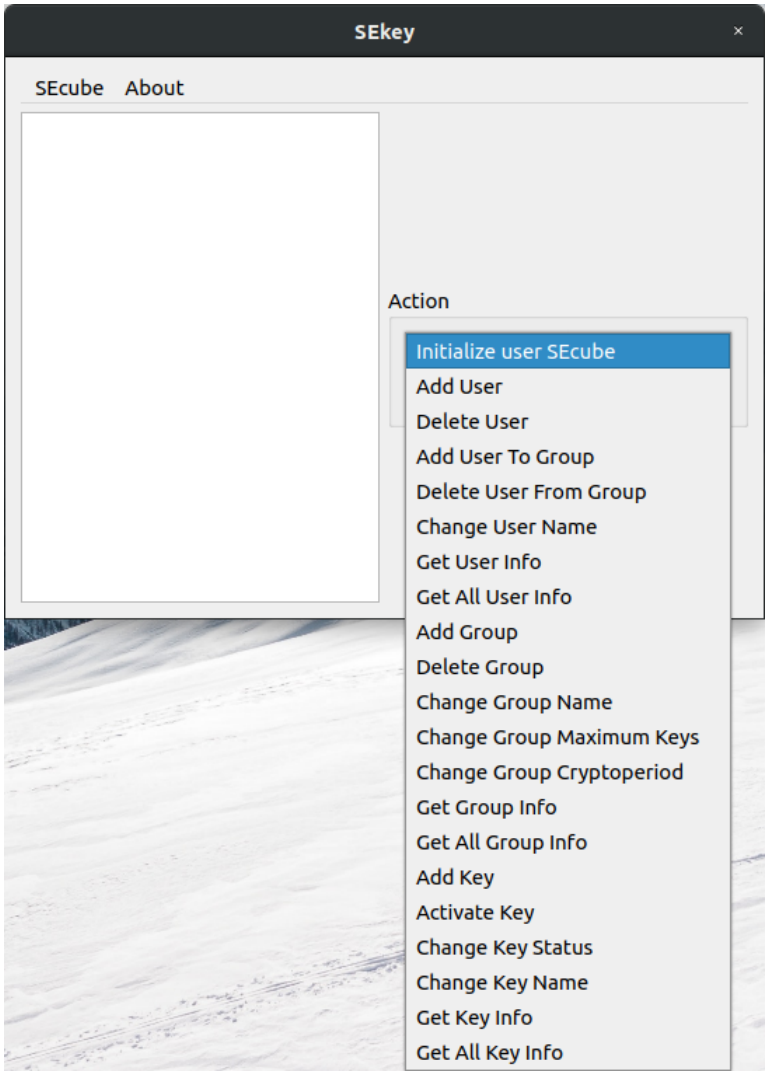

Figure 4: The list of actions that can be executed by the **SE*key*™** administrator.

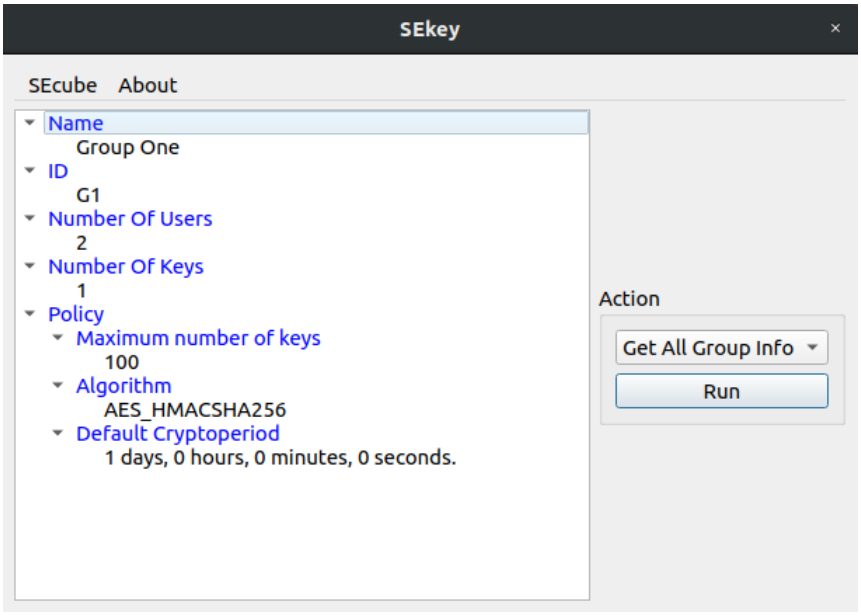

Figure 5: The result of the 'Get All Group Info' action performed by the administrator.