SEfile-cli

Project Documentation

Release: August 30th, 2019

Proprietary Notice

The following document offers information, which is subject to the terms and conditions described hereafter.
While care has been taken in preparing this document, some typographical errors, error or omissions may have occurred. We reserve the right to make changes to the content and information described herein or update such information at any time without notice. The opinions expressed are in good faith and while every care has been taken in preparing this document, some typographical errors, error or omissions may have occurred. We reserve the right to make changes to the content and information described herein or update such information at any time without notice. The opinion expressed are in good faith and while every care has been taken in preparing this document.

Authors

Daniele CASTRO danielecastro@hotmail.it
Nicoló MAUNERO *(CINI Cybersecurity Natinal Lab)* nicolo.maunero@consorzio-cini.it
Paolo PRINETTO *(President, CINI)* paolo.prinetto@polito.it
Antonio VARRIALE *(Managing Director, Blu5 Labs Ltd)* av@blu5labs.eu

Trademarks

Disclaimer

Contents

1 Features

This chapter presents the features of SEfile-cli Command Line Interface tool. The idea is to create a powerful tool that allows SEfile™ APIs to be used directly from linux bash or windows command prompt or within a script file without writing a single line of C code. It is also a very well-done usage example for those who wants to learn how to use SEfile™ APIs since it is very well commented:

- SEfile-cli supports many commands
- Each command does task that are mosto probably asked to be done by these APIs
- Each command supports a certain number of options
- Depending on the command some options are mandatory
- Not mandatory data options will be asked at runtime
- Code is particularly well commented for those who wants to learn how to use these APIs
- In each function prototype there is a doxygen comment explaining each single parameter

2 What is SE*file*™

SE*file*™ is a software library made to interact with the **SE***file*™ firmware to be flash on the **SE***cube*™ microcontroller. This firmware and library allow to use the crypto-engine peripheral on the **SE***cube*™ microcontroller to crypt files on the PC filesystem. It is a library which exploits the hardware key management exposed from APIs Level L1 and other functionalities from the **SE***cube*™ device. It has been developed having in mind the needs to ensure both simplicity of usage and security for data at rest: it allows secure storage, retrieve and usage of information that could not be trusted if stored elsewhere, e.g., any personal computer, or cloud service provider. More information on **SE***file*™ can be found here [1] Conceptually, **SE***file*™ targets any user that, by moving inside a secure environment, wants to perform basic operation on regular files. It must be pointed out that all encryption functionalities are demanded to the secure device in their entirety. In addition, **SE***file*™ does not expose to the host device details about what, or where it is reading/writing data: thus, the host OS, which might be untrusted, is totally unaware of what it is writing.

3 How to Compile the Project

In order to download the code visit the **SE***cube*™ website[2].
The project is a simple CMake one. You run CMake in the project directory and after you run Make. At the end of the compilation you will find the SEfile-cli executable in the root directory of the project. Here is a small script that can be run to compile this project in Linux:

```
cd SEfile-cli
cmake .
make
./SEfiel-cli
```

Some CMake-ready IDE like JetBrains[3] or CLion[4] can automatically recognise the CMakeLists.txt and compile the program without edit any setting. Alternatively, CMake can be run from a console and, once a Make file is generated, any IDE can compile everything.

4 Code Architecture

4.1 Global Architecture

The code is structured in three levels

- Command Line Interface (CLI) C functions
- Wrapper C function
- **SE***file*™ APIs

It is a CMake development project.

4.2 Command Line Interface (CLI) C functions

The Command Line Interface functions simply implements the functions that can be requested by the CLI interface with commands and parameters. They are:

[1] https://www.secube.eu
[2] https://www.secube.eu
[3] https://www.jetbrains.com
[4] https://www.ietbrains.com/clion/

```
void list (char * peripheral, char * password, char * directory)
```

This function shows a list of decrypted file names of encrypted files in a given directory

```
void wrcff (char * peripheral, char * password, char * file_path
   , char * cipher_file_path)
```

This function writes a cipher file starting from a binary file: it crypts the content of the binary file into a cipher one.

```
void wrcfs (char * peripheral, char * password, char * string,
   char * cipher_file_path)
```

This function writes a cipher file from a string: it crypts the content of the string into a cipher file.

```
void wrffc (char * peripheral, char * password, char * file_path
   , char * cipher_file_path)
```

This function writes a file from a cipher one: decrypts the content of the cypher file into a binary file.

```
void wrsfc (char * peripheral, char * password, char *
   cipher_file_path)
```

This function writes a string from a cipher file: decrypts the content of the cypher file into a string.
All these functions can be found in “SEfile-cli.c” and “SEfile-cli.h”.

4.3 Wrapper C Functions

The Wrapper functions are a middleware between the APIs and the CLI. They are:

```
se3_disco_it choose_devices (char * drive)
```

This function chooses a device given an ASCII capital letter ex: D, E in Windows or the mount point in linux. In Linux only already mounted devices will be shown.

```
void close_device (se3_session * s)
```

This function closes the SEcube USB connection. directory with decrypted file names.

```
se3_session init_device (char * password, se3_disco_it it)
```

This function handshakes the SEcube USB connection with the PC.

```
void list_cipher_files_in_directory (char * path)
```

This function lists all the encrypted files in the directory with decrypted file names.

```
SEFILE_FHANDLE open_cipher_file (se3_session * s, char *
   file_path, int32_t mode, int32_t creation)
```

This function opens the encrypted file.

```
se3_disco_it show_and_choose_devices (void )
```

This function prints a list of available devices ex: D, E in Windows or the mount point in linux. In Linux only already mounted devices will be shown.

```
void write_binary_file_from_cipher_file (se3_session * s, FILE *
    fd, SEFILE_FHANDLE * sefile_file)
```

This function reads an encrypted file and writes a decrypted version.

```
int write_buffer_from_cipher_file (se3_session * s, uint8_t *
   buffer, int buffer_len, SEFILE_FHANDLE * sefile_file)
```

This function reads an encrypted file and writes a decrypted buffer.

```
void write_cipher_file_from_buffer (se3_session * s, uint8_t *
   buffer, int buffer_len, SEFILE_FHANDLE * sefile_file)
```

This function reads a binary uint8_t buffer and writes an encrypted file.

```
void write_cipher_file_from_file (se3_session * s, FILE * fd,
   SEFILE_FHANDLE * sefile_file)
```

This function reads a binary file and writes an encrypted version.
All these functions can be found in “wrapper.c” and “wrapper.h”.

5 Command Line Interface (CLI)

Command Line Interface has been designed as simple as possible and in pure Linux style. No options abbreviations are available till now. Typing

```
SEfile-cli --help
```

will display a message well explaining how to properly use the program with many usage examples.

5.1 Usage

There are six commands and six options. Depending on the chosen command some options are mandatory. Not mandatory data options will be asked at runtime.
Program Commands

- *list*: list cipher files with decrypted names in a path
- *wrcff*: writes a cipher file from a file
- *wrcfs*: writes a cipher file from a string
- *wrffc*: writes a file from a cipher one
- *wrsfc*: writes a string from a cipher file
- *–help*: prints usage informations

Command Options

- *-pe*: stands for ’peripheral’, drive letter on Windows or partition path in Linux
- *-i*: stands for ’input file path/string’
- *-o*: stands for ’output file path’
- *-c*: stands for ’cipher file path’
- *-pa*: stands for ’password’
- *-d*: stands for ’directory path to list’

Usage examples:

```
SEfile-cli wrcfs -pa test -i "Hello world!" -c cipher_file_out.
   txt
```

On Windows:

```
SEfile-cli wrcff -pe D -pa test -i text_file_in.txt -c
   cipher_file_out.txt
```

On Linux:

```
SEfile-cli list -pa test -d /home/User
```

```
SEfile-cli wrffc -pe /mnt/sdb1 -pa test -o text_file_out.txt -c
   cipher_file_out.txt
```