Password Manager

Project for the courses Advanced Operating Systems and Embedded Systems

Authors:

Rosevinay Yelluri

Arlind Rufi

Jan Fischer

Contents

Specification

The program will provide a command line interface through the serial port printf/scanf and allow inputting and retrieving passwords that will need to be stored encrypted on the internal flash. Note that the driver for writing on the internal flash has been written in a previous project by other students, and it can be found here: https://github.com/goofy91/FileSystem

Miosix

Board name

User Manual?

The user interface

Writing to the flash

The data by the Password Manager is saved on the flash. It must be considered that the executable code (namely the operating system) is stored there as well. So one must be careful not to interfere with the data of the operating system. The Password Manager uses two functions to write (storeData) and read (loadData) the data from the flash. Both functions don't use parameters but the attributes of the PasswordManager class. For accessing the flash the memory layout described in table 1 is used. Here the standard start address is used as an example but it is valid for any other address with the corresponding offsets.

Address	Content	Comment
0x080F8000	'P'	Identification string
0x080F8001	'W'	"
0x080F8002	'M'	"
0x080F8003	0x00	"
0x080F8004	numOfPass	Short, which means that it occupies two bytes in the memory
0x080F8006	encryptedData	From here the flash contains WPTuples and the checksum of the password manager, which means all the user's stored password data
0x080F8007	"	Note: all this data is encrypted
...	...	...
EndOfData(= EOD)	0xFF	EOD = 0x080F8006+numOfPass*64+16 Because sizeof(WPTuple) = 64 and sizeof(checksum) = 16
Afterwards	0xFF	Area reserved for new passwords
...	0xFF	"
0x080FFFFF	0xFF	"

Table 1: Flash memory layout

For identification a string with content "PWM" is written at the beginning. In this way we can be sure that there is valid data in the flash and that there is no coincidentally valid data which doesn't make sense.

The loadData function

The storeData function

Address calculation

Encryption

Meeting Protocols

1. **Meeting on the 28. January 2016**
   Encrypting already implemented in miosix (use available functions)
   Make it simple as short time; Use static own defined data structure (struct) to store (website, password) tuples, and master password as well (e.g. 32 KiB on Flash). A password could consist of 32 bytes.
   On Flash one can only write zeros/0 (to write ones/1 the whole sector must be erased – all values set to 0xFF)
   Erasing Flash destroys it; so erase only when necessary
   It is possible to use cin and cout as well as string for the ui and convert it for flash storing to char*
   Read can be implemented by using memcpy from miosix (copy from flash to ram with struct pointer).
   Most C++ library functions are usable in miosix and may be used.

2. **Meeting on the 12. February 2016**

Glossary

Standard start address – used to write/read from flash. It is the last address in the flash with 32 KiB of space behind. This way it is as far away from the operating system as possible. The address is 0x080F8000. It is defined in Passwordmanager.h as STANDARD_ADDRESS.

WPTuple – (or WebsitePasswordTuple) is a struct defined in PasswordManager.h. It is used to store a website and the corresponding password.