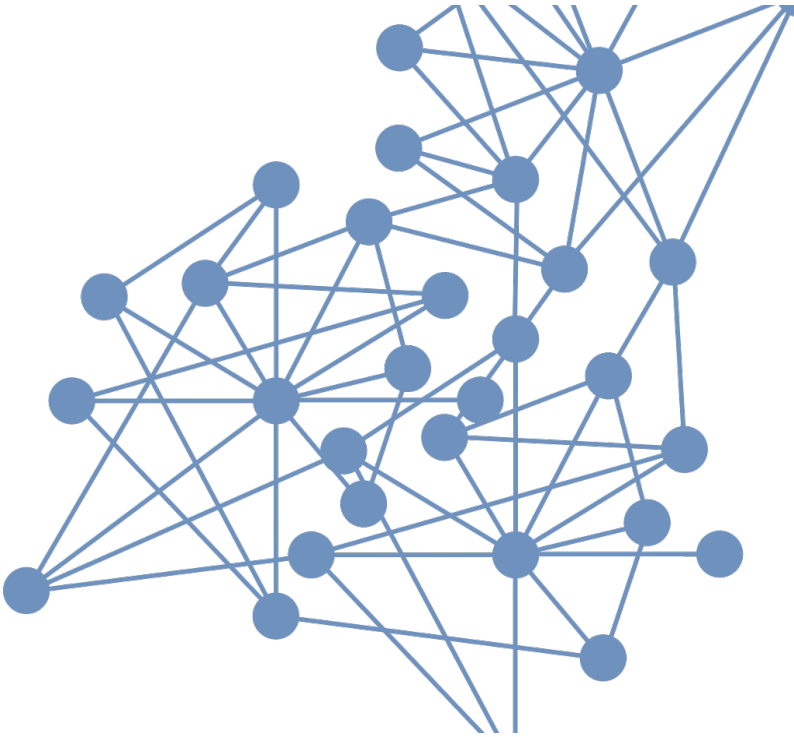
BettyCrypt
Bettysoft Research

September 2016 | English



Introduction

Never before the importance of good and easy to use encryption was higher than today. People want to secure their digital selfs with less complications. They are not interested in any algorithms but they want them to work correctly.

This is where BettyCrypt sets on. It is an open source encryption library with well known algorithms written in C++. At Bettysoft, BettyCrypt forms the heart of SecretFolder, but everyone who wants to use the library for its own projects is allowed to do.

Due to it is open source, the code of BettyCrypt can be analyzed by anyone who wants to. A big community of code-analyzing people is good, because errors in code are found faster and the library becomes better.

The following document explains the architecture of BettyCrypt with all its classes.

Library architecture

General

Written in C++, BettyCrypt offers both, good performance and great opportunity in designing. To make the library easy extendable, Bettysoft uses the idea behind polymorphism – an abstract master class with virtual methods which are set to null. Just the child classes define these methods.
In its newest version BettyCrypt fully supports all major platforms, due to the usage of elements from C++11/14.

Crypt – The master class

The master class of BettyCrypt offers the two most important methods of the whole library.

> *virtual bool decrypt(...) = 0* and *virtual bool encrypt(...) = 0* are virtual and null. Every child class of *Crypt* has to define these methods otherwise the program will not compile.

Besides these two methods, there are three more static methods that are important for BettyCrypt.

> *static unsigned long long int fileSize(...)* returns the file size of a given file.
> *static int readPadding(...)* returns the padding that has to be added to the input data while encryption. That is important for block cipher algorithms because not all input data fit perfectly into the blocks, so there is a padding left over.
> *static std::string readCryptVersion(...)* returns the type of algorithm and its version a file is encrypted with.

The child classes

The child classes are the different encryption modules, for example AES. Each of them has to define the decryption and encryption method of *Crypt*. So every child class forms its own encryption and would work on its own without any polymorphism.

Polymorphism

So, if every encryption could work on its own, why is polymorphism used in the BettyCrypt library?

The fact that users can choose the algorithm they want to use to secure their files with, complicates the architecture of the encryption library. To avoid this, BettyCrypt uses polymorphism.

The abstract master class cannot be used to create objects of itself but to create pointers. These pointers can point to an object of any child class of *Crypt*. With this knowledge encrypting and decrypting data is very easy to realize with less code and the library is even easy to extend with new algorithms.

The update function

The update function offers a great opportunity to let every programmer decide on his own to implement a (lambda-) function that can handle the the current progress of the de- or encryption.

The programmer has to implement this function. If he does not want to handle the current progress, he simply can put an empty lambda-function as the class methods parameter.

AES

The Advanced Encryption Standard is a block cipher with a block size of 16 bytes. BettyCrypt integrates it with 128 Bit and 10 rounds.

Future view

For the near future the AES with 256 Bit is planned to be integrated in BettyCrypt.