Erika Dains

CS480

April 7, 2014

PasswordKeeper README

Running the program:

* The file to run is Driver.java
  + There are 5 other .java files to the program: Cryption, PassDeletion, PassEntry, PassGeneration & PassRetrieval.
  + There is also the PasswordKeeper.txt file, which will be the file that stores your encrypted passwords. This file needs to be in the same folder as the program for proper running.
* Running Driver.java will open an application which will first ask you to enter a password. The password for this program is “thisismypassword” (no quotes).
* Once you type in the password you can click on the navigation buttons on the left side.
  + Generate new password
    - Will automatically generate a password, although you have a button that you can click to regenerate the password as many times as you want
    - When you’ve found one you like, click Save Password button.
      * Enter the necessary information and click save.
  + Retrieve a password
    - Enter the necessary information and click search, matching passwords, if any, will show up in text box below.
  + Delete a password
    - Enter the necessary information and click search, results will show up in below many, select the password item you want to delete (useful for when there are two matches), and click delete button. You will receive a successfully deleted message.

Design Decisions

1) Password generation: This program generates 16 character passwords by looping through a random number generator 16 times. This number generated by the random number generator corresponds to a place in an array of letters (upper and lower case), numbers, and special characters (!@#$%^&\*()?). A password such as this is difficult to crack in a program such as our lab 3 ones because it does not use any words from the dictionary, and does not have any pattern as to where a letter, number, or special character may be.

2) Password storage: I used the AES symmetric key encryption to encrypt the text in the PasswordKeeper file. I chose symmetric key because it is faster for encrypting/decrypting files and better for large file sizes. The key is secure because it is not hard coded into the code, instead it is created each time the program is run by applying a SHA hash to the application master password and converting that hash into the secret key.

4) Ask user for a password: The MD5 hash of the master password is hard coded into the code, however the password itself is not. Once the user enters the password in the application, their entry is hashed using MD5 and checked against the hash that is hard coded – if it matches, the navigation buttons on the side of the program are enabled and the user can navigate through the application as explained above. If it doesn’t match, the user is given an error message and asked to try again.