Project I
CS 6238: Secure Computer Systems
SecLogin: Stronger authentication with keystroke features

Due Date: 10/21/2014

Maximum Number of Participants per team: Two

The goal of this project is to implement a more secure authentication scheme (we hope so) using techniques described in the password hardening paper that was discussed in class. To simplify the project, we assume that a user types a password at a client machine. A trusted component at the client records the typed password and captures keystroke feature values. The client securely sends the password and keystroke feature values to a server where the actual authentication is done. Thus, the server implements the functionality outlined below for a given user *U*.

Initialization:

The initialization will require following steps:

1. Select a 160 bit prime value *q* and choose a random H_{pwd} value such that $H_{pwd} < q$. Also, choose a value *h* for the size of the history file (*h* is the number of recent feature value vectors that are stored in the file).
2. If number of distinguishing features is *m*, then select a random polynomial *f* of degree *m*-1.
3. Using the polynomial *f*, generate a Instruction table such that all α and β values are valid, i.e using either entry, a correct H_{pwd} can be calculated. To calculate α and β values, use the password and the formula given in paper.
4. Create a fixed size *History* file. Select its size, pad it and encrypt it with H_{pwd}. Use suitable redundancy to ensure so you can find out when encryption is successful.

A login request arrives at the server from a client. In your implementation, assume that the server reads a request from a test file that you create. Such a request will include a password text and the feature values. On receiving a login request, the server must execute the following steps to authenticate the client.

1. Select appropriate value i.e., α_i or β_i from the instruction table based on the feature values $\phi(i)$ in the request.
2. Extract the value selected from instruction table using password supplied in the request.
3. Calculate the x_i, y_i coordinates based on the α_i or β_i values selected from the instruction table using the formula given in the paper.
4. Calculate value of H_{pwd} using the coordinates using polynomial interpolation.
5. Decrypt the history file using H_{pwd}. If decryption is correct then you have calculated the correct H_{pwd} and hence the login request is from the legitimate user.
6. Once you have authenticated the user, you will add the feature values for the current login in the history file and encrypt the history file again with a new H_{pwd}.
7. Select a new random polynomial *f'* and using it generate a new instruction table.
8. Compute each α and β value using password as key.

Some Important points:

- The history file should be saved to disk only after it is encrypted. Decrypted version of file should not be stored on the disk. Also the size of the file should remain constant; if the size is less than the selected size then the file must be padded before encrypting it.
- Random polynomial *f* of degree *n*-1 is defined as :
  $f(x) = hpwd + a_1x + a_2x^2 .... + a_{n-1}x^{n-1}$ where the values a_1 to a_{n-1} are chosen randomly.
- In calculation of α and β, G_{pwd} is a keyed hash function. You are free to choose the hash function you want to use and how you use the key supplied to it to calculate the keyed hash value.
- You can implement the project in either C/C++ or Java.

For grading, you must have a functioning implementation to receive credit. Partial credit may be awarded when some of the steps are not completely implemented. In addition to source code (it should be well commented and checked for any code vulnerabilities), you should provide information that shows that all required features are implemented.

You will be required to demo your stronger authentication implementation to the TA. The TA may provide a test file containing password and feature values for a sequence of login requests that your server code will need to process. The format of the test file and an example of it will be made available before the submission deadline. You will schedule a time for a demo after the project is completed and submitted via t-square.