College of Computing, Georgia Institute of Technology

CS6238 Secure Computer Systems

SecLogin: Stronger Authentication With Keystroke Features

Project 1

Chidong Xue

902885919

cxue8 [at] gatech.edu

M.S. in Information Secuirty

2014.10.20

Introduction

This is project is an implementation of a more secure authentication scheme, described in the paper Password Hardening Based on Keystroke Dynamics. A standalone Java program is implemented, which accepts user inputted user name and password via console, and typing feature values via file, to determine user authentication result. User is also able to create or initialize an account, with choosing user name and setting password.

This reports includes discussions about the design and implementation of the project, potential integrations and evaluations. An simple demo of the program with screenshots is attached in appendix as well.

Design and Implementation

FILE STRUCTURE

The program has two types of input: (1) console input, including user name and password; (2) file input, including a feature value file, and a threshold value file. The later two files are provided before running the program, and stored under ./ress (resources). The feature value file is called “input”, and its format is <sequence number of user> <username> <15 feature values>, with each user occupying one line. The threshold file has exactly 15 lines, with each line containing an integer to describe the threshold for corresponding feature.

While the program has two types of output file for each user: (1) instruction table file and (2) encrypted history file. All these file for all users are stored under ./users, with username as part of the file name. For example, for user with user name “james”, there are two files stored for this account under ./users, namely “james_table.txt” and “james_history.txt”. Table file contains exactly 15 lines, describing the instructional table. Each line contains two values that are separated by comma, with lefthand side being α and righthand side being β. Fixed-size history file is encrypted with harden password, and it cannot be directly read.

INITIALIZATION

On the start of the project, program ask user to login (1) or create account (2), or exit (3). User select by typing 1 or 2 or 3 in console. After user choose 2, initialization process starts. User is asked to input desired username and 8 bit password, if username is taken or password inputted is not 8 bit, user will be asked to input again.

After user inputing valid username and 8 bit password, a random H_{pwd} value is generated, such that $H_{pwd} < q$, with q being a 160 bit random prime value. A random polynomial function of degree 14 is also generated, by randomly generating 14 more random coefficients, and using H_{pwd} as the first coefficient, such that $f(0) = H_{pwd}$.

From the polynomial function, 30 points are calculated, that all these points are on the function. i.e. $\{y_{ai}^0, y_{ai}^i\}_{1 \le i \le m}$ are calculated that all the points $\{(2i, y_{ai}^0), (2i+1, y_{ai}^j)\}_{1 \le i \le m}$ are on the function. Then use the function described in the paper, as follow, to calculate α and β. G function is implemented

$$\alpha_{ai} = y_{ai}^0 + G_{pwda}(2i) \bmod q$$

$$\beta_{ai} = y_{ai}^1 + G_{pwda}(2i+1) \bmod q$$

with SHA1. Pwd_a is used as key in hashing process, and this is actually implemented in the way that, before inputing value into SHA1 function, an additional operation is performed between Pwd_a and 2i or 2i+1. Afterwards, calculated α and β values are then stored in the table file for the corresponding user for future usage.

History file is then initially created. Its structure is:
<Key Used To Recognize>
<Username>
<Successful Login Times>
<Feature Values Of Last 8 Successful Logins> (8 lines)
Here, the key/token to recognize successful login is simply a string "THISISGOOD0987654321". Other more complicate token can be used in reality. Successful login times is initialized to zero, and also feature values of last 8 successful logins are all set to 0. Afterwards, this file is padded to size 800 bytes with '\0'. History file is then encrypted with AES with $H_{pwd,}$ which is modified by padding or trimming to make it a valid 128 bit AES key, and stored.

An "Account Created" is prompted to notify user in the end of the initialization process.

LOGIN
On the start of the program, user select 1, to start login mode. User need to input user name and password. System first check the existence of the username based on stored files, then start authentication process. The login phase is basically the reverse process of initialization. Inputted password will be used together with feature values read from file to generate the discrete nodes, and then generate polynomial to find H_{pwd}. The formulas used here are:

$$(x_i, y_i) = \begin{cases} (2i, \alpha_{ai} - G_{pwd'}(2i) \bmod q) & if \phi_i(a,l) < t_i \\ (2i+1, \beta_{ai} - G_{pwd'}(2i+1) \bmod q) & if \phi_i(a,l) \geq t_i \end{cases}$$

Each feature value will be compared with the threshold value read from threshold file to determine whether α or β value should be utilized in the computation. Simple data structure of multi-dimensional array plays a role in storing the data for relatively efficient retrieving. After node has been generated from the formulas, polynomial interpolation can be performed to determine the hpwd based on these nodes. Even though the paper suggests a Langerange coefficient for interpolation, I chose the Newton form for convenience. (Due to the unisolvence theorem, this form shall give the same results). Apache common math library is utilized.

After get the Newton form of interpolation polynomial, intersection of the polynomial in the bi-dimensional coordinate system can thus be calculated. Thus, H_{pwd}' is gained. Use this H_{pwd}' as key (padding or trimming may be applied to make sure it is a valid 128 bit AES key), to decrypt corresponding history file. After decryption, if first line in the file exactly matches the token described above, then the system recognize this login attempt as successful one.

After successful login, history file is updated, including updating successful login in times, and adding this time feature values to past records. Then it is encrypted again and stored. While, new

polynomial function is generate, and new instruction table is constructed, with similar process as it is in initialization process.

Only difference is that, after three successful logins, program will begin to update its instructional table. Because less than three successful logins, we cannot have enough data to calculate mean and standard deviation. After after three successful logins, we start to find distinguishing features. For distinguishing features, either α or β value will be garbaged in the table. The formulas to find distinguishing feature is: $|\mu_{ai} - t_i| > k * \sigma_{ai}$, k value here used is 0.04, as discussed in the paper. Last new table will be stored, replacing the old one.

A successful notification will then be shown to user in console.

Integration

This project is now a standalone program, but is has the potential to be extended with other program, to form a whole harden password authentication system. The potential program can be cooperated may includes (1) a keystroke record program, to replace feature file input; (2) an GUI, for example a webpage or a windows application, to replace console; (3) a client side, and use this program as a server side, etc.

A concrete example could be: use this program (with modification) as a backend server, to work with a webpage with keystroke functionality. This could be useful for web most applications.

Appendix: Example

Note: (1) There is a static boolean called 'DEBUG' in the program, if it is set true, then all debug information will be printed along with regular information. For regular usage, it should be set to false, but it is set to true here in these screenshot.

CREATE ACCOUNT

```
*** Login (1) or Create Account (2) or Exit (3): 2
Please Enter User Name: lee
Please Enter Password: 12312312
HPWED 11030515069842759391
Account Created!

*** Login (1) or Create Account (2) or Exit (3):
```

Account is created, with Hpwd = 11030515069842759391.

LOGIN - CORRECT, 1ST TIME

```
*** Login (1) or Create Account (2) or Exit (3): 1
User Name: lee
Password: 12312312
Verifying......
***DEBUG***: Calculated Hpwd: 11030515069842759391
***DEBUG***: decrypted history file:
THISISGOOD0987654321
lee
0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
***DEBUG***:
*Harden Authentication Off: Less than 3 successful logins, insufficient data
***DEBUG***: new file:
THISISGOOD0987654321
lee
1
11 99 39 70 40 90 13 10 88 98 31 98 4 11 19
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Authentication Result: true

*** Login (1) or Create Account (2) or Exit (3):
```

As you can see, the above shown the original history table for “lee”, which is successfully decrypted, and below is the update table. Note, because there are less than 3 successful logins, we will not start to update the instructional table.

It shows: “Authentication Result: true”

LOGIN - CORRECT, AFTER 3 TIMES

```
*** Login (1) or Create Account (2) or Exit (3): 1
User Name: lee
Password: 12312312
Verifying......
***DEBUG***: Calculated Hpwd: 11030515069842759391
***DEBUG***: decrypted history file:
THISISGOOD0987654321
lee
6
31 89 40 70 30 80 23 15 89 82 31 82 34 23 19
31 89 40 70 30 80 23 15 89 92 39 92 24 13 39
31 89 40 70 30 80 23 15 89 92 39 92 24 13 39
11 99 39 70 40 90 13 10 88 98 31 98 4 11 19
11 99 39 70 40 90 13 10 88 98 31 98 4 11 19
11 99 39 70 40 90 13 10 88 98 31 98 4 11 19
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
***DEBUG***:
*Harden Authentication On.
***DEBUG***: new file:
THISISGOOD0987654321
lee
7
33 81 42 70 30 80 23 15 89 82 31 82 34 23 19
31 89 40 70 30 80 23 15 89 82 31 82 34 23 19
31 89 40 70 30 80 23 15 89 92 39 92 24 13 39
31 89 40 70 30 80 23 15 89 92 39 92 24 13 39
11 99 39 70 40 90 13 10 88 98 31 98 4 11 19
11 99 39 70 40 90 13 10 88 98 31 98 4 11 19
11 99 39 70 40 90 13 10 88 98 31 98 4 11 19
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Authentication Result: true

*** Login (1) or Create Account (2) or Exit (3):
```

After 3 times, harden password scheme is working, and table are updated with some garbage values.

LOGIN - WRONG PWD

```
*** Login (1) or Create Account (2) or Exit (3): 1
User Name: notexist
*User Name Do Not Exsit. Please Enter Again: lee
Password: 12345678
Verifying......
***DEBUG***: Calculated Hpwd: 12343731916072668675787982064706903743066790764O
***DEBUG***: decrypted history file:
??A##T?6?F[??J?W????????????,???86???`??A?j????)???!0&r?6?:?    !UFY??u`z?`?R)??w???3??a??ob0??PWO????l?&??@F???5wKOu0<??????}7?/]
Authentication Result: false

*** Login (1) or Create Account (2) or Exit (3):
```

Wrong Hpwd is generated, and the corresponding history file cannot be correctly decrypted.

LOGIN - CORRECT PWD, BUT VERY DIFFERENT FEATURE

```
*Harden Authentication On.
***DEBUG***: new file:
THISISGOOD0987654321
lee
7
33 81 42 70 30 80 23 15 89 82 31 82 34 23 19
31 89 40 70 30 80 23 15 89 82 31 82 34 23 19
31 89 40 70 30 80 23 15 89 92 39 92 24 13 39
31 89 40 70 30 80 23 15 89 92 39 92 24 13 39
11 99 39 70 40 90 13 10 88 98 31 98 4 11 19
11 99 39 70 40 90 13 10 88 98 31 98 4 11 19
11 99 39 70 40 90 13 10 88 98 31 98 4 11 19
0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
Authentication Result: true

*** Login (1) or Create Account (2) or Exit (3): 1
User Name: lee
Password: 123123
*Please Enter a 8 bit password.
Password: 12312312
Verifying......
***DEBUG***: Calculated Hpwd: -1100253240149218111478468699
***DEBUG***: decrypted history file:
f
???l??  ]??????????2Yb?!?!?O4?!?????2X??k??d????`??j?,???D}???{~???????NWC n?G??''??????I?????S??3J?]!???FVr?????zO/,A?3#ThhYXza?Q
9?^A?Z$????AR?SU??f???x[}_s4??q?1>Xf:???!??{b??g?R??av??????P???um6???S?O???????5~A??????:??'???????w+z?M?`?????9wF?,l??I!3??1??!
r?R
???N    ????wo??????3j?c)??l?n>{?A?N??#_dp?????0OHE??m???G??#=?'-n}!??z???^2&???>rD?XX?????5??>????6?Lennzm
??????????R?O
Authentication Result: false
```

With correct pwd, but feature value is changed (to some degree), user cannot login to the account.