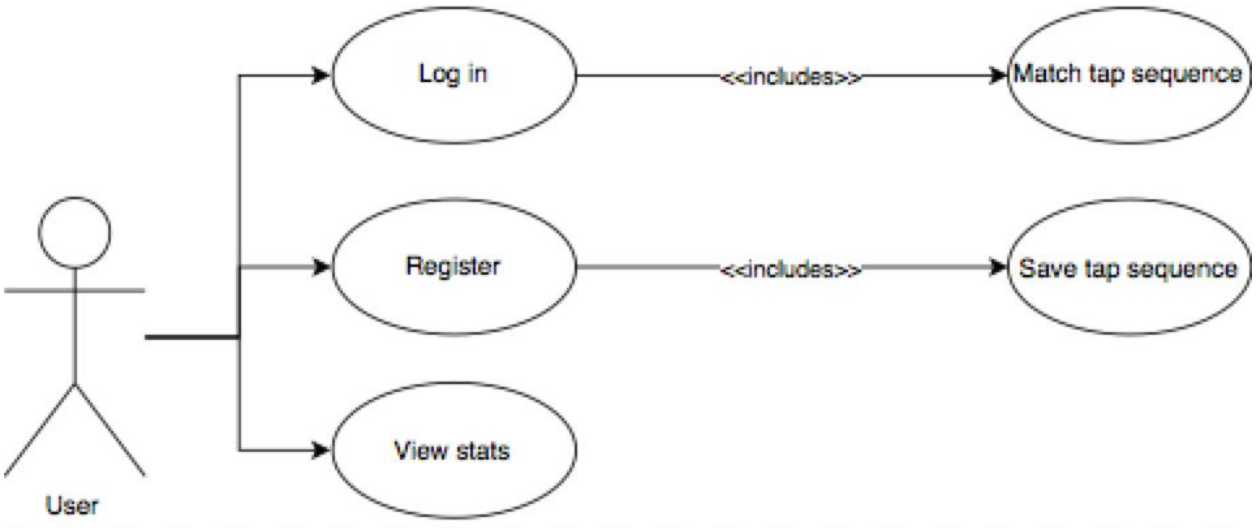
Pistachio

Deliverable: System Requirements

Team: Pistachio

Members: Cameron Dziurgot, Luke Brodowski, Travis Moretz

Submission Date: 7th October 2015



Functional Requirements:

This Android application will be designed for testing the accuracy and feasibility of using a complex passkey stored as vectors for securing an application.

A user will save a passkey that will be a series of screen presses where the duration of the press, area of screen contacted, duration between presses, and location of press will be stored and saved as a vector. The process of saving a passkey will require them to enter the same passkey 3 times. The system will take the average of the 3 attempts and set this to the users passkey. This will give a more accurate representation of how the user enters the passkey, as repeating the same passkey will be virtually impossible, but the average will be more repeatable for the user.

The user will then be able to attempt to reenter their passkey that the application will validate as being correct or incorrect to a certain degree of accuracy.

The user will be asked if the attempt was meant to be a valid attempt or invalid attempt to keep statistics on the accuracy of the system.

User Stories

1. A user will be able to create a new passkey, during which they will be asked to enter the same passkey three times to get the average of the three.
2. A user will be able to enter a passkey attempt and have the application compare it to a saved passkey to a certain degree of confidence.
3. When a user enters the application and chooses to enter a passkey, a button with “Start” will be on the screen with a rectangular box next to the “Start” button to for entering the passkey.
4. After a user presses the “Start” button to begin entering a passkey the “Start” button will change to a “Stop” button for the user to stop the passkey attempt recording.

5. A user will be able to see statistics on properly entered and falsely entered passkey attempts.
6. When viewing the user statistics the user can close the statistics screen by pressing the close button.

User Story Pre/Post Conditions

1. **Setting a passkey**
    a. Precondition: The application must not have a passkey set
    b. Postcondition: The application will now have a passkey to test
2. **Testing a passkey**
    a. Precondition: The user must have a user account and passkey saved to it
    b. Postcondition: The system authenticates
3. **Starting a passkey attempt**
    a. Precondition: The user is logged into the application and selected to enter a passkey attempt
    b. Postcondition: The start button will change to a stop button and the system will start recording the screen inputs in the input area of the screen
4. **Stopping a Passkey attempt**
    a. Precondition: the user must have started a passkey attempt by pressing the start button
    b. Postcondition: The system will display if they attempt was a valid match to the saved passkey or not a match
5. **Viewing User statistics after valid passkey entered**
    a. Precondition: The user must have entered a matching passkey attempt to their saved passkey and have answered the query on if it was a valid attempt.
    b. Postcondition: Once the user is done viewing the statistics they will be able to get to their home screen by pressing the back button
6. **Closing the User Statistics Screen**
    a. Precondition: The user must have entered a matching passkey attempt, the query must have been answered, and the statistics must be displayed
    b. Postcondition: The user will be back at their home screen.

Largest User Story

The largest user story that can be broken down into multiple user stories would be the setting up the initial tap sequence. A user would need to enter their tap sequence a couple of times so we can get an accurate average. This story has already been broken down into different user stories.

Non Functional Requirements

1. **Product requirements**
    a. The product should be able to prove, or disprove, the feasibility of tapping sequences as a viable version of biometric authentication.

2. **Usability requirements**
    a. Within 5 minutes of use, a user should be able to understand how to navigate the different functionalities of the application.
    b. Help texts will always be displayed since explanations will not be longer than one sentence because of simplicity of device.

3. **Performance requirements**
    a. The application speed will be high/fast. The most computationally challenging process in the system will accessing a sequence and running a comparison against the input sequence. This should take an insignificant small amount of time.
    b. All functionality in the system is self contained, no outside system interaction is required aside from retrieving stored sequences from device memory. This makes runtime for the system quicker and more efficient.

4. **Space requirements**
    a. Stored sequences will use an insignificant amount of memory.
    b. The applications simplicity allows it to take up an inconsequential amount of space on an Android device. When running, RAM usage of the system will be trivial.

5. **Reliability requirements**
    a. Failure rate should be extremely low provided that the system is simple in its tasks and does not require any input from systems outside of the Android OS.
    b. System should be available at any point an Android device is on, since functionality of system does not require any outside input besides user driven events (e.g. entering a sequence, pressing a button).

6. **Portability requirements**
    a. Application should be able to be downloaded and installed on any Andriod system, and will run on any system with touch screen capabilities.

7. **Delivery requirements**
    a. Documentation for this application will be done to par with the CS410 course project outline for deliverables.
    b. Version control will be through GitHub. Standard code documentation will be expected for easily readable code.

8. **Implementation requirements**
    a. The application will be downloaded and run an installation on a device. At which point, it will allocate space for storing sequences.
    b. The application can then be opened on the device and all functions will be accessible to a user.
    c. There will be a way to offload collected data from a device onto a computer.

9. **Interoperability requirements**
    a. This application will be designed only to communicate with the Android OS. No other system interaction is required in order for the device to function.
    b. Data on sequence and statistics will be stored in the Android memory, allowing outside programs to access the data without communicating directly to the system.

10. **Ethical requirements**
    a. This application will be used to verify a new way to protect a user's private data.

11. **Legislative requirements**
    a. While this application is not meant to be used as an authentication method for a device, the legality of this system will fall under the same laws as other forms of biometric authentication. These laws may vary depending on state or region.

12. **Privacy requirements**
    a. While the application is meant to prove authentication feasibility, the device does not keep user sequence's private. This is because even with the data of a user's sequence, a sequence should not be replicable.

Glossary

Degree of confidence: A calculated observation that is a good estimate for the accuracy of an averaged parameter.

False acceptance: Likelihood that the system will incorrectly accept an access attempt by an unauthorized user.

Passkey: A unique series of screen taps or presses and time between the taps and presses that a user will save to the applications as their own unique identifier.

Passkey attempt: After a user has saved a passkey for their profile an attempt will be where the user tries to re-enter the same passkey to compare against the saved passkey.

Profile: A sequence/passkey combination stored on a device.

Sequence: Series of screen taps and presses.

Tap-Box: The box that will detect the tapping sequences from users.

User statistics: Statistics for passkey attempts.