SECURICON

Idil Sukas
Binghamton University
isukas1@binghamton.edu

Okan Gul
Binghamton University
ogul1@binghamton.edu

Nilay Altun
Binghamton University
naltun1@binghamton.edu

ABSTRACT

Our objective, to create a system that enables additional security layer via wireless connection between two personal devices; cell-phone and computer. Desktop application for computer side in NetBeans IDE and android application for cell-phone side in Android SDK are implemented. We as a SecurIcon team, learn how to establish a wireless connection between two applications by socket programming approach. We tried to make log-in process easier, choose password as an image password, due to the fact that added more security to our SecurIcon project. The project is successfully completed; users are able to send their image passwords to desktop application from android application.

1. INTRODUCTION

The applications and websites that we commonly use today, send passwords via text message as an additional security layer. For example, online banking systems mostly use that kind of approach to be make sure that the customer matches with his/her account number. The companies that need to be very secure are using cell-phones as a proof of customer's identity. Due to the fact that we thought that a data in our personal cell-phones can be used as a password.

The data that we are looking as a password can be an image. Since, the target of the project is to provide protection; image passwords also hide the existence of password, which makes it even more protected. Due to the fact that, we worked on implementing a system that provides its users additional security layer by sending password as an image to targeted application.

What will our final project will provide its users? The first utility that SecurIcon project provides its users, an additional security layer that user who wants to access desktop application, need to send password from his/her personal cell-phone which has specified IP address. The second, the users no need to memorize passwords anymore, since the password is an image and kept in cell-phone, user don't need to memorize the password, he/she just need to select from photo library which is easier to remember. Once again, it is very quick and easy access to a desktop application because the user does not need the type the password so it makes access quicker and easier.

2. DESIGN OF SECURICON

2.1 Desktop Application

The Frame of "Netbeans IDE" was used in desktop application. The Frame of NetBeans is a class of the "java.awt" package. A frame is resizable and movable window that has a title bar and close buttons, maximizes and minimizes. Desktop application has two different frames. First frame includes signup and login pages. Second frame belongs the real application page. The real application page should have any design. In our application, the real application is very simple. The both first and second frame is a container, but the first frame has some components like label, button, and text field. We tried to design a clear user interface and every user can easily understand how the desktop application works. As user interface is clear and simple so the program is preferable. For signup page, we used a file chooser to upload the image as password. Additionally, there is menu bar in the frame. We designed a button for socket connection in the menu bar. We needed to database and used NetBeans local database for the SecurIcon project. As a design, database keeps a basic user's information into the columns.

2.2 Android Application

As a team, this was the first time that we worked on an android application. It took a little bit time to learn about the new environment. Android SDK is used for designing SecurIcon android application. Our main goal about application's design to make it easy to use. There is nothing complicated, every user can easily understand how the application works in just few seconds. Clear user interface makes it very easy to use. After opening desktop application, "password" input text appears on the screen, the user need to start SecurIcon android application for password selection. A file chooser opens automatically, the only thing that user need to do, to select password which is specified before and click on "Upload" button. Connections and specified new path on computer works on background. Due to the fact that user do not need to follow complicated steps to enter his/her password.

Figure 1. Sign up page

3. IMPLEMENTATION OF SECURICON

3.1 Desktop Application

Desktop application was implemented in NetBeans IDE. For desktop application, a Frame is necessary. Firstly, we had to create the Frame class for the user interface. The user interface

has buttons, text fields, labels and menu bar. We used some methods of a Frame class like "*setTitle(String value), setSize(int row,int col), setVisible(boolean mode)*" to set the window features. Users have to use "signup" and "login options to control the program. "Signup" option has two fields. Users have to enter a unique username and select password with using a file chooser as image from computer to sign up. "Login" option has a system entrance and menu bar. The system entrance includes username and password fields and some buttons. The "login" button was implemented to enter the system (real application page). For the server side of socket programming, the program starts to listen after click the start connection button in menu bar. Users can enter password with using this way. They have to enter a username and send password to enter the system. The image comparing was implemented in the "login" button. Images are checked byte by byte in the "login" button event. User's password (image) that is sent from Android application must to be exactly the same with database.

In database, we created a local NetBeans database to keep user's information. There are two columns: "username" and "password". "Username" is a primary key for SecurIcon database, and it have to be unique. Database is not allowed to same two username. The type of that column is "char". "Password" has to be image. In the "password" column, there can be same two images as password. The type of "password" column is "blob". "Blob" type is used for keep image. The both "password" and "username" column cannot be empty.

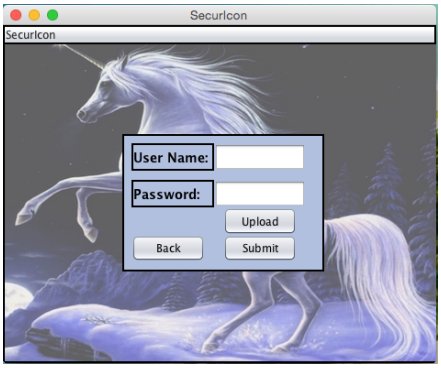

Figure 2. Log in page

3.2 Android Application

For android side of the SecurIcon project, implemented user interface and client side of socket programming. The user interface has "Upload image" button, "Select from library" button, an image viewer and an alert dialog. While server side is listening for a package to receive with in specified port, user need to click the screen to select his/her password from photo library. After selecting the image from library it can be any format such as jpg, bmp, png, etc. the image viewer will be activated with click event listener and will show the selected image on the screen. User needs to click "Upload image" button as last step for sending the image to password field in desktop application. If the password delivered without any error, status alert dialog window will give information such as by saying, "delivered".

In the background, the program gets selected image from image viewer and sends it server side from client side in specified IP address and port number. In the set-up process, IP address of computer that desktop application works needed to be entered to create a connection between two applications. IP address can change based on location, due to the fact that it is important to update new IP address to establish a connection otherwise the android application will not be able to send image via connection.

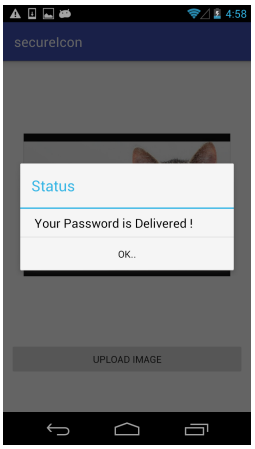

Figure 3. Android Status

3.3 Socket Programming

Socket programming is the challenging part of our project. We need to connect the two platforms, which are phone and computer to make our security layer. As you know our users choose a unique icon when they sign up our application (server side), they choose the icon and use that icon to get access to our system. When they need to access they simply open up our phone application and choose icon from their photo library and with an easy send button they send it to the desktop application to reach their account in the system. But what is happening on the backside of this process? We reach our goal of connection with socket programming.

Socket programming is basically gives the users to send any kind of data from one platform to another. Simply you can think it is building a water pipe between water spring and your house to get drinking water. For the server side which is our desktop application we build up socket object with ServerSocket servsock = new ServerSocket(8000); line. As you see we build up a socket with the port number of 8000. This Java code helps us to build up a listening server with the computer's static IP and the server starts to listen this port for clients.

For the client side we use sock = new Socket ("198.255.147.138", 8000); to build up a client side socket object to send the password icon. As you see we build up our socket with the information of the desktop application which are static IP number "198.255.147.138" and port number "8000". When we choose the image and push the send button this object is building up and connect with the server to send the proper data. The security layer which we add is the need of two programs be in a same wireless network, if they aren't in the same network our connection is never been build up. So user connect the same network and choose the IP number and then send the password to the desktop application these steps add our deliverable lots of security which is our main goal.

At last phone application gets the image data from the specified path which user choose his icon from his phone then client code put it in a buffered stream and send it to the server for the further processing. Again server side simply got the data and stores it to the specified path in the computer to control it with the database for giving user the authentication.

Figure 4. Start Connection

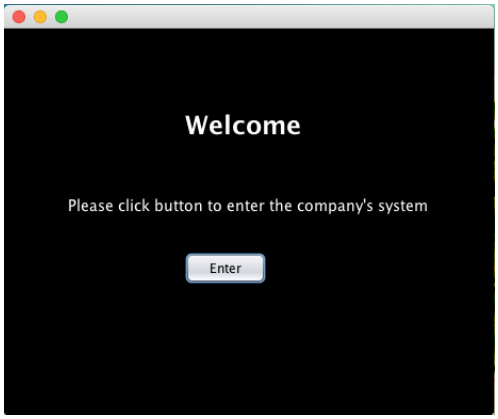

Figure 5. Real Application Page

4. EVALUATION

Our securIcon project implementation solves the security problem for companies for giving authentication to the secure company system. The project creates an external layer for other companies systems and it can be used as an embedded system. The usage areas are many and various, every company can be able to use it very easily for their internal controlling such as their employees access of the internal systems.

Our improvements, with the use of different kind of platform such as Android, Java as a team reached our goal to build up a complex system. With the help of socket programming we got familiar with the network programming and learn every aspects of the connection between different applications in separate devices.

4.1 PROS

After completing all the design and implementation parts, team members moved on testing processes. The result is pretty satisfying. Our aim was to make program efficient, execution time is very short. In testing process, we tried to consider every test cases that we can think, for all the test cases, execution time took between three to five seconds which is shorter than we expected.

We had a chance to test our program to other students in class, thus it really helped to get feedback from different approaches. Such as we planned at the beginning of the semester, it made access very easy and quick. Generally, the users of online banking in current system, they need to wait for receiving text messages and it sometimes fails. If it fails or exceeds the given time period, the users need to redo same process again and again which is very time-consuming. With the help of our project, they do not need anymore to wait for text messages, which is slower.

We searched for the related works on Internet and any other scholars. We did not come across with the design that we offer right now. So, it makes our project unique approach to create extra security layer.

4.2 CONS

Besides positive feature of the project that we implemented, there are also assets that we can improve in the future to make it more efficient. One of the negative aspects, IP addresses needs to be specified beforehand. Since IP address based on the location of computer, its IP address is changeable. Every time the IP address changes, android applications' will need new address to be able to establish connection. In the future, we will work on to make it automatic.

Our project is an external layer and it should be embedded to other applications. Due to the fact that, every company might ask change according to their own preferences. However, providing a support service will easy solve this issue.

The user also needs to make sure that password photos should match exactly. It should be in the same size, for example if the user tries to send it via Facebook messenger, it will change images' resolution and it will not match with the specified password. Due to the fact that, users need to be very careful when sending their passwords over an application.

5. LIMITATIONS

We have different kind of limitations in our different stages of project. The first limitation we come across was the building up a port on a wireless device. When we try to open a port on our home wireless devices it is very hard to get through the protocols and build up a specified port on our router and also modem. We try to get through it with using Binghamton University's wireless network, which they already build up a port 8000 which we, control the system and start to use it for our project.

We have also meet same difficulties on socket programming phases, again we tackle on protocols, for this specific one android didn't give us the authentication of building up the socket on the main thread of the OS. With a detailed research we found a library, which help programmers to build up child threads for their own usage. So Asynctask [6] help SecurIcon client side code to done it's all work on a child thread, after this solution we get through this problem and continue to our path to reach our goal.

At last we have some difficulties while using local database on Netbeans IDE. We need to download Derby.Jar for configuration of the database, also have some difficulties for sending the local database to other computers but solve it with the sending proper files and building database on Netbeans on needed computers.

6. CONCLUSIONS

In this paper, we present the design, analysis and implementation of the SecurIcon project. We used "NetBeans IDE" and "Android Studio" for implementation. To keep user's information, we selected "NetBeans" local database. Briefly, our system that provides its users additional security layer by sending password as an image to targeted application. Our program has two different deliverables: Desktop application and Android application, and also we needed to connection between these applications. Because of that, we used socket programming, and we connected them each other.

7. ACKNOWLEDGMENTS

Thanks for our professor Mo Sha to his endless help to our project to make it alive. His help is crucial for our success, and also with his help we can try and work our application with his labs android phone to make our deliverable bug free and stable. Also thanks our classes TA Prashant Kadam for his understanding attitude to us with our sometimes easy and sometimes hard questions, he is always be helpful to us and make our project better with his solutions to our problems. At last thanks to our all team with their effort on the whole process to bring our project to life, as individuals they work hard without any doubt.

8. REFERENCES

[1] Socket Programming Tutorial. http://www.tutorialspoint.com/java/java_networking.htm .

[2] Android Design. https://developer.android.com/design/

[3] JavaCv Tutorial. http://opencv-java-tutorials.readthedocs.io/en/latest/

[4] NetBeans Desktop Application. https://netbeans.org/kb/trails/matisse.html

[5] Android Studio. http://developer.android.com/tools/studio/index.html

[6] AsyncTask Android. http://developer.android.com/reference/android/os/AsyncTask.html