Securing Mobile Apps: Application Lock with Image Authentication for Android OS (ALISA)

Fathul Arif Bin Kamarudin
Universiti Kuala Lumpur
UniKL MIIT
Kuala Lumpur
fathulkb@gmail.com

Dalilah binti Abdullah
Universiti Kuala Lumpur
UniKL MIIT
Kuala Lumpur
dalilah@unikl.edu.my

Wan Hazimah binti Wan Ismail
Universiti Kuala Lumpur
UniKL MIIT
Kuala Lumpur
wanhazimah@unikl.edu.my

***Abstract*— Even though accessibility to Android mobile phone device is strictly restricted to only its rightful owner, there is still a possibility for someone to gain access and use the application in the Android without any restriction, especially if that Android is being lent to someone who is close to the owner such as family and friend. The aim of this project is to develop an application lock with image authentication for Android OS. The app will be developed using Android Studio and Java language programming which inevitably incorporate a sequence of images that is set exclusively by the authorized user as a form of authentication. The app will be able to prompt the user to provide the right sequence of images before allowing access to the application that has been lock by it. This project utilized the recognition based technique for the graphical password implementation by using story scheme as its main algorithm. The prototype features a randomization of images for every instance of image passcode page viewing, replacing the alphanumerical password with graphical password and able to unlock and lock application using the image sequence. Based on the testing results, the prototype is able to lock all application for every android version but unable to achieve positive result on android version 5.0 and above for the unlocking aspect. The prototype unable to pass integrity and confidentiality aspect of security testing but succeed in the availability aspect. This prototype can be further improved by implementing a hybrid of graphical password technique and through the usage of up to date Java programming language in Android Studio.**

Keywords— Mobile Application, Image Sequence Authentication, Security, Android Application Package (APK)

I. Introduction

Authentication is simply a process that validates the authority of the user to access the object of desire; the most vital role in system security in which it makes sure that sensitive application is only accessible to the intended person only. This is because a normal person has natural needs for privacy [1] However, it is difficult to prevent such application from being randomly access by others due to the lack of ability to lock sensitive applications by the legitimate user. In fact, with the way the world works nowadays through the constant need for internet and technologies, privacy has slowly becoming something unobtainable. In Malaysia, there is an act that specifically being enacted for the purpose of protecting people's privacy right; The Personal Data Protection Act 2010 (PDPA). The act ensures the protection for personal data from being modify without the authorization of the user [2] Nowadays, many organizations have found the means to explore the ways and to push functionality to mobile devices; given rise to the increasing amount of sensitive data onto the devices. Due to the sudden spike in the number of mobile devices, their technical capabilities, and their use for information transactions attached to back-end enterprise applications, mobile platforms have become an increasingly attractive security target. Plus, the significance of mobile application security is related to the fact that many mobile services are often unsecured and exposed, creating potential confidentiality breaches from unauthorized access [3]. Even with the number of option for handling access and authentication, there are still some drawbacks present. That is why focusing on securing the mobile application takes a precedence role by using a specifically design app with the ability to restrict the access of applications that the user might consider to be private.

This project is to developed android based application with the purpose to ensure all of the applications installed in the Android mobile phone device are safe from being access by unauthorized user. Furthermore, this application implements image sequence for authentication and image randomization instead of just password which theoretically more secure. The application allows a sequence of unique images in JPG format to be selected by the admin to make a graphical password from a single set of one unique image to another, each derived from different category of image type ranging from everyday stuff. The applications' interface provides 3 x 4 grids of push-button containing one unique image for each 10 image passcode respectively; accepts password input up to four as minimum and maximum number.

II. Background of Research

A. Problem: The need of Privacy for Android Device

Various applications can be run on Android OS devices for the benefit of the authorized user as their rightful owner. These applications are accessible for the user with administrator right because the system recognizes their authenticity and authorization to make changes on the application. However, even if the application in the Android is accessible only to its rightful owner, there is a possibility for someone to gain access and use the application in the Android without restriction, especially if that device is being lent to someone who is close to the owner such as family and friend. However, it is hard to make sure whether the applications installed in the mobile phone are only accessible by the legitimate user. This is because there is lacking of access policies, security awareness or lack of usage in access restriction-related tools [4].That is why for a standard user, specific application with the ability to lock other applications by the legitimate user is needed to prevent unauthorized access.

B. Why Graphical Password Password is better

Nowadays, there are many methods presented by developers to secure authentication. However, even with the ever growing numbers of options for authentication, alphanumeric password still remains the most popular and widely used method of identification. Even if it is true, there is still a draw back in this method. According to Birget, [5] he described this drawback as "password problems" because the requirements for password that needs to be fulfilled are conflicting. First, password should be easy to memorize and the execution of the authentication protocol should be fast and easily done by human. Second, password should be secure in a way that it appears to be random and difficult to guess, changed periodically, and different on each platform of the same user.

Furthermore, it cannot be written down or unencrypted. This proves to be difficult to achieve due to its conflicting requirements. This is true because most of the users will not be able to remember difficult and random password. Not to mention every password must be different on every account. That is why statistic shows that 65% of workers use identical password either for different applications or services [9]. This may lead to critical security-related problem in the future. By replacing the alphanumeric password with graphical password, the password problem may be solved [4]. The idea of graphical password is to let the user click either with mouse, tap or stylus on a few chosen regions in an image that appears on the screen. In order to log in, the user simply has to click in the same regions again [6]. Most graphical password systems are based on either recognition or cued recall. In recognition-based systems the user must recognize previously chosen images from a larger group of distractor images whereas in cued recall password systems, users must click on several previously chosen areas in an image; cued by viewing the image. Both types of systems may have memory advantages over alphanumeric passwords [12].

C. Choosing the Right Technique

There are several different types of techniques and method that can be used to implement graphical password. The techniques can be categorized into three main techniques which are recognition-based, pure recall-based and cued-recall based approaches.

1) Recognition Based technique

This technique involves the user to choose pictures, icons or symbols from a collection of images. In authentication process, the users need to recognize their registration choice among a set of candidates. Based on the research, 90% of users can still remember their passwords after even after 63 days [10]. One of the algorithms that are used in the recognition based technique is Story. The Story requires one round of authentication, but with password pictures that are conduct in a sequence of number of unique images that makes a story to enhance memorability. When users authenticate, users have to click the password pictures. Then, the Story needs the users to remember the order of images. By doing so, it results to a much harder time for users who are not applied their story to guide the image selection. Studies show that, of all incorrect password entries in Story, over 80% of them contained all the correct images, but with incorrect order [7]. This implies that making a story should be emphasizing to users to help them memorize easier.

2) Pure recall-based technique

This technique which can also be referred as draw-metric system allows the user to recall an outline drawing on a grid that they invented or selected during the registration phase. In other word, users draw their password on a grid on a blank canvas. Memorability is difficult in case of recall. This is because it is difficult for retrieval of memory to be done without any reminders or cues [11]. One of the algorithms used in this technique is Syukri algorithm. This algorithm proposed the authentication to be conducted by having user to draw their signature using mouse or stylus. The users will be asked to draw their signature first with mouse and then the system will extract the signature space and either enlarges or scale-down signature areas. The verification stage first takes the user input, and then extracts the parameters of the user's signature which later verify by the system using geometric average. The main advantages of this approach are signatures are hard to fake and there is no need to memorize one's signature [11].

3) Cued Recall Based technique

This technique is related to identifying specific locations. It proposed a framework of reminder, hints and gesture that help the users to reproduce their passwords or help users to make a reproduction of password more accurate. This will increase the memorability as it is easier to memorize than pure recall based technique. The users are provided with an image so that they can choose points arbitrarily by clicking in the presented image as a password. For login, the user simply clicks on the right click points in the correct order [11]. One of the algorithms used in cued recall based technique is Persuasive Cued Click-Points. This algorithm proposed the users to select a click-point within the viewport. At the time of password creation, the images are slightly shaded except for a random small viewport area positioned on the image. Users can then click on the "shuffle" button to reposition the viewport randomly until an ideal location is found by the user [8]. Table 1 shows the comparison between graphical password techniques.

Table 1: Summarization of Major Graphical Password Techniques

Technique	Algorithm	Usability		Effective Against Attack
		Authentication Process	Memorability	
Recognition based	Story	A sequence of number of unique images that makes a story. When users authenticate, users have to click the password pictures	Users need to remember the order of images. Need to emphasis on sequence more	Brute search, Dictionary attack
Pure Recall Based	Syukri	Draw signatures using mouse. Need a reliable signature recognition program	Very easy to remember , but hard to recognize	Brute search, Dictionary attack
Cued Recall Based	Persuasive Cued Click-Point	Proposed a framework of reminder, and gesture that help the users to reproduce their password	Help users to make a reproduction of password more accurate	Brute search, Dictionary Attack

III. METHODOLOGY

For this project, recognition based techniques was chosen to be implemented as it is the most balance technique out there in term of authentication process and memorability. However, the researcher also found out some of the neat features that can be implemented in the project such as inserting reminder, hints and gesture that can help the users to reproduce their passwords more accurately. The randomizer button is also nice feature that can be implemented in the project as it can lessen the possibility of guessing and shoulder surfing.

The project is developed using Android Studio with Java Language programming. The Android Studio software is mainly used to develop the project's application by using its flexible Gradle-based build system. For this project, the development starts off with brainstorming for initial requirements of features and functions as well as the design of the application. It is later developed to a proper system based on the phases present in the development process. The phases are consisted of, application development requirement, application design, and stage of development.

A. Application Development Requirement

This is the phase where information is gathered and the appropriate software needed in this project were also identified. This is a vital part in the development process because all requirements must be met and ready to be used in order to proceed to the next phase. The specific software that been used to developed this application is Android Studio version 2.2.2 where it is an Official Integrated Development Environment (IDE) for Android app development and support Java Programming language. OPPO version 4.2.2 was used as the Android device Platform in order to test the application for this project during the development process.

B. Application Design

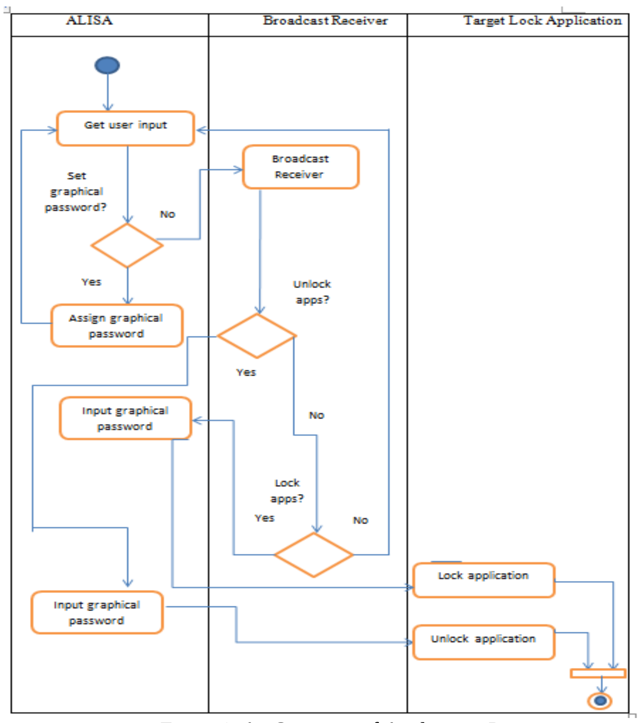


Figure 1: An Overview of Application Process

During the design process, the application interface is first sketched to give a bit of perspective on what to expect in the design of its interface before the real implementation of it is conducted. Besides that, the overview of the whole process of the application was also modeled using the activity diagram as depicted in Fig. 1.

According to Fig. 1, the application started by obtaining the user input first, i.e. accessing to the first page of the prototype. Next the user is given a choice whether to set graphical password or not. If the user agree, the application obtained the user graphical password. If the user disagree, the broadcast receiver perceives the user intention whether to unlock or lock the application. Should the user intended to lock the app, the broadcast receiver transferred the intention to the application for obtaining the user's graphical password input. The application is unlock once the application receives the correct graphical password. Should the user prefer to lock the application instead, the broadcast receiver transferred the intention to the application for the graphical password input from the user. The application is lock once it receives the correct graphical password from the user.

C. *Application Development*

In this project, all the design and development process took place in Android Studio using its application module. The module contains three main folder which acts as project's key source files. The folders are; manifest (contain the AndroidManifest.xml file), java (contains the Java source code files) and res (contains all non-code resources, such as XML layouts, UI strings, and bitmap images.).

From the main page, authorized user can set up their image passcode, change it or disable the app by disabling the image passcode. The disabling function requires the authorized user to re-enter their old image passcode before it is activated. Locking function can be started by tapping on the Start Locking button. Once this function is activated, the applock process will be started in the android background device. The locking session then will be in active mode and once the authorized user turn off the screen of their device, the locking session will take over the background service; ultimately locking the device and its applications. Any changes in image passcode required by the authorized user can be done by tapping on Change Passcode button. This function if activated will prompt the authorized user to enter the old image passcode before entering a new one. Fig. 2 below shows the main page of the prototype's interface.

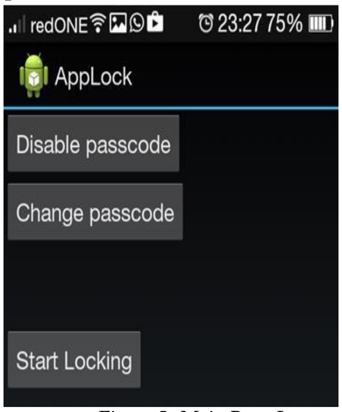
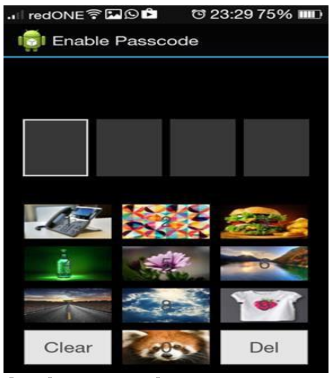


Figure 2: Main Page Layout

The image password accepts a minimum and a maximum of four images as passcode. The app's image password interface provides 3 x 4 grids of push-button in width and height respectively containing one unique image on each ten



push- button grid. The image buttons consist of 10 different images with 74x74 of width and height in size respectively. The size is chosen specifically as to carter with reasonable space in the frame layout and for aesthetic purpose. Each push-button holds another unique number that acts as an id key for the image in its respective grid. The image buttons are randomly shuffled each time the authorized user accesses the given page. From there, the authorized user is prompted to enter their chosen graphical password by creating a sequence of images password based on the given graphical button. As the shuffling process occurs, the authorized user simply enters the previously chosen images through the correct sequence; allowing the authorized user to authenticate their identity. This page is prompted for the first time when the authorized user installed the prototype and every time the authorized user accesses the prototype.

The authorized user simply chooses the passcode from a mixture of few categories to make a story. The categories must be different and are originated from categories that depict our everyday life such as food, cars, pets, etc. [7]. Only when the sequences of images are correct, then the authorized user is authenticated. For this project, there are nine buttons with nine different images for each button displayed at any given time. Fig. 3 shows the image password page of the prototype's interface.

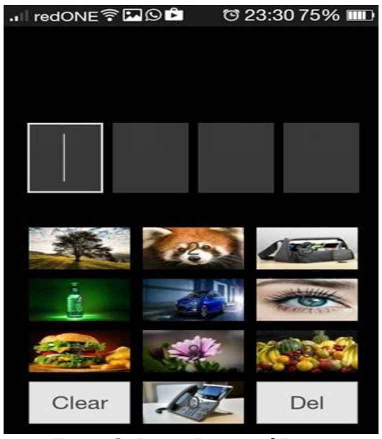


Figure 3: Image Password Layouts

The lock app page is prompted once the authorized user has set the image sequence passcode and starts the locking function from the main page before turning off the android screen. Once the authorized user turn on their android device screen, this page is prompted. The authorized user then simply enters the designated image sequence for authentication. If the sequence is correct, the authorized user is granted the permission to go to home screen; where applications installed are accessible. If it is the opposite, permission to go to the home screen is not granted. For this application, the users are not limited to a number of tries. This page is functioned to block all access from unwanted user to the application installed in the device as shown in Fig. 4.

Figure 4: Lock App Layout

IV. RESULTS AND FINDING

For this project, there are two methods which has been implemented. The two methods are consist of Functional Testing and Security Testing.

A. Functional Testing

The testing is performed by providing the respondent with the prototype to test its functionality based on the requirement and scenario given. The respondent consist of five students ranging from 15 to 25 years old and two government workers of age 30 and above. This is taking into consideration the appropriate age when android device becomes popular to certain age group. Based on the data conducted by China Marketing Blog [13], people of aged 20 to 40 are the main Android users. Thus, for this project, the target respondent is primary focus on the corresponding age group.

Each of the android versions tested on has different API level starting from Jelly Bean (4.2), KitKat (4.4), Lollipop (5.0) and Marshmallow (6.0). The samples of functional test case given are taken from the result of testing conducted by developer, a university student and a school teacher for summarization purposed.

1) Lock and Unlock Succesful Rate

Under the functional testing, a test is conducted to specifically determine the rate of lock and unlock for the App Lock page to be successful. This consistency is required to have on both aspects in order to determine whether the prototype is considered accomplished its objectives or the opposites. For this test, the same testers are required to access the Lock App page 10 times via turning off the android screen after enabling the locking function. Each of the android versions tested on has different API level starting from Jelly Bean (4.2), KitKat (4.4), Lollipop (5.0) and Marshmallow (6.0). Figure below shows the result taken from the test case for Lock App successful rate under the functional testing.

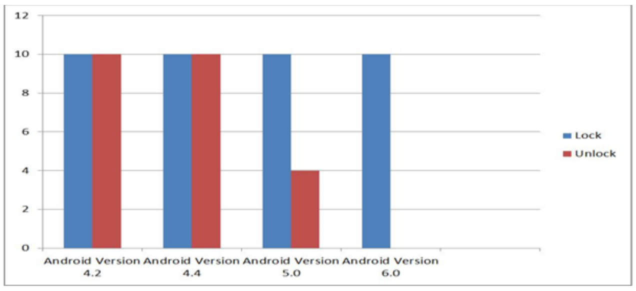


Figure 5: The Lock and unlock Successful Rate

Based on Fig. 5, the chart shows that locking application yielded positive result whereas unlocking application yielded negative result when tested with 10 trials from various android version. Android version 4.2 and version 4.4 managed to yield 100% successful rate on both aspect of unlocking and locking. This is because the prototype was developed using those two versions as its main testing platform and changes and modification in coding either due to deprecated methods or codes usage in Android Studio also revolve around those two versions.

Despite that, the chart shows that android version 5.0 to 6.0 manage to yield 100% success rate on locking aspect. However, that is not the case with unlocking aspect for android version 5.0 as it only successfully unlock the application three times out of ten whereas version 6.0 literally unsuccessful to do so. The inability of higher android version to successfully unlock the application on 100% rate is probably due to the deprecation of old methods and codes that were previously used in Android Studio older version and later obsolete in the latest version. This caused random conflict in its service background handling; resulting to screen freeze and preventing unlocking to take place. As the prototype was developed based on version 4.2 as its main platform and the development process was referred to tutorial on older version of Android Studio, higher version may require different approach in method implementation and further knowledge in Android Studio programming is needed. This shows that the app is not properly developed due to deprecation of certain methods in Android Studio.

2) Frequency of Image Passcode Randomization

Under the functional testing, a test is conducted to specifically determine the rate of image randomizing for the Image Passcode page to occur. This is because the shuffling or randomizing of the images plays a vital part to further increase the difficulty of non-authorized user to recognize the images displayed on the Image Passcode page. For this test, the same testers are required to access the Image Passcode page 10 times from any angles; i.e. tapping change passcode or locking or disabling the prototype. Each of the android versions tested on has different API level starting from Jelly Bean (4.2), KitKat (4.4), Lollipop (5.0) and Marshmallow (6.0). Figure below shows the result taken from the test case for Image Passcode randomization rate under the functional testing.

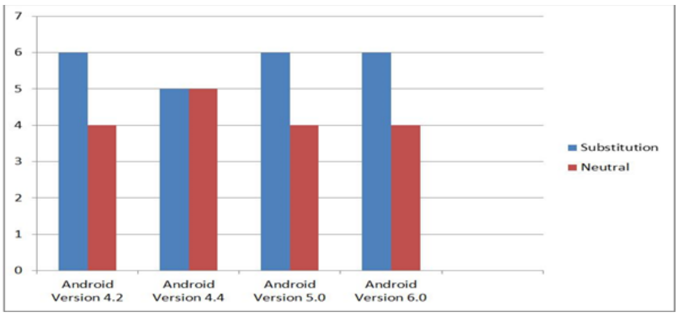


Figure 6: The Frequency of Image Randomizing

Based on the Fig. 6, the result shows the substitution of images per one button occurred six times whereas the image stays the same for four times when tested with 10 trials. This is roughly the same for android version 4.2 and above with only Android versions 4.4 have an equal rate. This is primary contributed by the array listing the image in the Random class. In the Android Studio, the button contains three different images from the same categories of everyday stuff. The array is the one that sorted the image and list it for Random class to randomize it. This shows that randomization did occur and it can be increased by adding more images in the array list.

B. *Security Testing: Confidentiality, Integrity and Availabality*

The security testing is a method to examine and analyses the security aspect of Application Lock App. The way it executes is by performing the testing using the security testing methodology. The testing is run on two levels; device and application level. At the device level, there are two ways in which the application shall be tested:

- With Android device running in a factory default or normal mode
- With Android device running in a rooted mode

On the application level, there are two ways in which it shall be tested:

- Application running on the device (to take benefits of touch related features)
- Application running on the emulator (to ease the task of testing using wider screen of desktop or laptop)

For this project, the testing is performed with android device running in normal mode for device level and application running on the device for application level. This limitation is applied to the testers. Developer however follows all the approaches except for using emulator due to performance constraint.

The CIA stands for Confidentiality, Integrity and Availability. This is a famous security model which designed to guide policies for information security within an organization; or in this case the prototype of this project. That is why this model is adopted into the security testing; to cover all aspect of security that should be addressed as a whole by including the aspect on confidentiality, integrity and availability.

Table 2 below shows the requirement to run the security testing.

Description	To determine the security aspect based on CIA model
Setup	Apply: Resetting, turn off and on screen rapidly or end all function
Expected result	Unlocking application using those methods is unsuccessful
Actual result	Unlocking application using resetting and end all method is successful

Table 2: Test requirement for Security Capability

Table 3 below shows the summarization of the result taken from the test case for security capability of prototype under the security testing.

Security Model	Specification	Result
Confidentiality	Android application inaccessible to unauthorized user	Fail
Integrity	Android application retain its originality	Fail
Availability	Android application accessible to authorized user	Pass

Table 3: Summarization for Security Capability

Based on Table 3, the result shows that only Availability meets the requirement in security model. This is because the prototype is available for the authorized user to use in any condition providing that the prototype is not uninstalled in the first place. Confidentiality and integrity however do not meet the security requirement. This is because any user can actually access the application even if the application has been lock by the prototype by simply performing either these two methods; resetting the android device or use end all function. These two bugs contradict both the security model of confidentiality and integrity. Once the unauthorized user manages to find a way to access the application, the confidentiality lost its purposed. As a result, the integrity of the application is also lost due to the meddling of unauthorized user; i.e. modify, uninstall, copy, move and so forth.

V. CONCLUSION

The research was conducted thoroughly for the duration of time regarding the authentication technique availability, its operability and how it involves with image password. There are myriad of methods that can implement graphical password. However, after enough materials gathered from the research, the authentication technique with image password had been chosen for this project. The development process was initiated to develop ALISA. The prototype was first designed to be able to lock and unlock each application individually with its application list support. However, due to severe deprecation of some methods essential for achieving this in Android Studio and lack of tutorial to learn the new method, the design was scrap halfway through development. Despite all that, a workaround has been found. Instead of locking and unlocking application individually, the prototype is designed to lock and unlock all application in one go. This design was able to push through to the final stage of the development. The testing involved several testers from a certain age group. The result shows the substitution of images per one button occurred six times whereas the image stays the same for four times when tested with 10 trials. This shows that randomization did occur and it can be increased by adding more images in the system library. The locking application yielded positive result with 10 trials whereas unlocking yielded negative result.

A. *Strength of ALISA*

- Able to lock and unlock all application
- Replace alphanumerical password with graphical passcode using image sequence for its authentication technique
- Image passcode randomization

B. *Weakness of ALISA*

- Bugs in resetting and end all function resulting to termination of prototype service
- Inconsistency with main function on higher API version of android
- Lack of proper coding usage due to deprecation in Android Studio

C. *Recommendation*

There are ways to improve this project due to some weakness contributing to some of the major flaws in the current prototype. Below is listed several recommendations for future enhancement.

- Lock and unlock each application individually instead of bundle.
- Apply more reliable technique for graphical password using hybrid.
- Use up-to-date programming language to develop better app in term of functionality and security.

REFERENCES

[1] Falkvinge, R. (4 September, 2013). Why Do We Need Privacy, Anyway? Retrieved 13 November, 2015, from PrivacyInternet Access: https://www.privateinternetaccess.com/blog/2013/09/why-do-we-need-privacy-anyway/

[2] KASS. (2015). Privacy Rights. Retrieved 13 November, 2015, from KASS: http://kass.com.my/data-protection/.

[3] Denim Group. (1 July, 2014). Denim Group. Retrieved 9 March, 2016, from Why is Mobile Application Security Important?: http://www.denimgroup.com/blog/denim_group/2014/07/application-security-important.html

[4] IT Security. (2013). Unauthorized Access. Retrieved 13 November, 2015, from TeleLink: http://itsecurity.telelink.com/unauthorized-access

[5] Birget, J. (December, 2007). The Graphical Passwords Project . Retrieved 13 November, 2015, from Clam.Rutgers.edu: http://clam.rutgers.edu/~birget/grPssw/

[6] Blonder, G. (1996). Graphical Password. United States: United States Patent 5559961.

[7] Davis, F., & Reiter, M. (2009). On user choice in graphical password schemes. Proceedings of the 13th Usenix Security Symposium. San Diego: Usenix Security Symposium.

[8] Chiasson, S., Oorschot, P. C., & Biddle, R. (2010). "Influencing users towards better passwords: Persuasive Cued Click-Points". Human Computer Interaction (HCI).

[9] Leyden, J. (18 April, 2003). Office Workers Give Away Passwords for a Cheap Pen. Retrieved 13 November, 2015, from The Register: http://passwordresearch.com/stats/statistic116.html

[10] Phen-Lan, L., Li-Tung, W., & Po, W. H. (2008). Graphical passwords using images with random tracks of geometric shapes. 2008 Congress on Images and Signal Processing.

[11] Ramanan, S., & S, B. J. (2014). A Survey on Different Graphical Password.International Journal of Innovative Research in Computer, 4.

[12] Wiedenbeck, S., Waters, J., Birget, J.-C., Brodskiy, A., & Memon, N. (2005). Authentication Using Graphical Passwords: Effects Of Tolerance And Image Choice. SOUPS '05 Proceedings of the 2005 symposium on Usable privacy and security, 1-12

[13] Samantha. (23 July, 2014). 5 Charts on Android Popularity in China. Retrieved 2 December, 2017, from China Marketing Blog: https://www.nanjingmarketinggroup.com/blog/china-mobile-phones/5-charts-android-popularity-china