Secure and Usable Authentication on Mobile Devices

Roland Schlöglhofer
KEBA AG
Gewerbepark Urfahr
A-4041 Linz, Austria
+43 732 7090 23310
slgh@keba.com

Johannes Sametinger
Johannes Kepler University
Altenbergerstraße 69
A-4040 Linz, Austria
+43 732 2468 4251
johannes.sametinger@jku.at

ABSTRACT
Mobile devices contain a multitude of sensitive data and provide access to even more data as well as services somewhere on the Internet. Even if only temporarily in the hands of non-entitled persons, privacy is at stake. Authentication protects against unauthorized usage. Today's operating systems of mobile devices offer authentication mechanisms. However, they are either vulnerable in some situations or not user friendly enough to be widely adopted. In this paper we suggest a novel authentication system which meets both the requirements of security and usability. For that purpose, we have analyzed existing authentication methods as well as targeting attacks. The resulting Android application SecureLock is a generic authentication system, which offers PIN and password, but also a property-based authentication method by means of NFC tags, and a novel image-based method called GesturePuzzle. The application has been evaluated and compared with other approaches for security and usability.

Categories and Subject Descriptors
D.4.6 [**Operating Systems**]: Security and protection—Access controls, authentication; D.2.0 [**Software Engineering**]: General; H.4 [**Information Systems Applications**]: Miscellaneous.

General Terms
Security, Human Factors.

Keywords
Mobile devices, security, authentication, lock screen, usability, Android.

1. INTRODUCTION
Mobile devices have captured the end users' Internet world. Smart phones and tablets are increasingly used for services on the Internet, such as mailing, browsing or socializing. Using such services involves the storage of sensitive data on these mobile devices, like contacts, mail messages, phone calls, short message texts, etc. These data have to be protected, be they for personal use only or for business use as well. Security mechanisms include the protection of information from disclosure and corruption. It is also important to keep the information accessible and productive to its intended users. Such mechanisms are quite mature on personal computers but less so on mobile devices. In addition, as the name suggests, mobile devices get carried around by their owners and may get lost or stolen much easier than their desktop counterparts. Once in the hand of an unauthorized user, sensitive data stored on the device as well as data on the Internet, that is being accessed with the device is at stake. Therefore, users of mobile devices typically install some form of access control, typically a PIN, i.e., a personal identification number that is a secret numeric password and has to be entered in order to access the device. Typically, PINs are rather short for usability reasons, often four-digit numbers resulting in ten thousand possible numbers. But short PINs often do not provide enough security for really sensitive data. Users could choose to use long and complex passwords, but would have to enter many characters with virtual keyboards on their touch screens, which is too cumbersome in most cases.

In this paper, we will introduce authentication mechanisms as well as attacks against them. We will then suggest an authentication mechanism for Android devices, compare it to existing solutions, and evaluate their security and usability. For our considerations we assume mobile devices with touch screens and virtual keyboards.

MoMM2012, 3-5 December, 2012, Bali, Indonesia.

2. AUTHENTICATION
Authentication is the process of determining whether a particular person or device should be allowed to access a system, an application, or specific data on a device. This process is an important basic security mechanism [1]. Authentication schemes can be broadly classified into three categories, knowledge, i.e., what we know, ownership, i.e., what we have, and inherence, i.e., what we are. We will describe these categories in more detail in subsequent sections.

2.1 Knowledge
The traditional username/password or PIN-based authentication scheme is an example of the type "what we know". Other examples include questions and answers as well as graphic passwords. The knowledge is used to authenticate at the device. The secret knowledge may get into wrong hands. Everyone having this knowledge is able to use it for authentication. Challenge-response authentication is also based on knowledge. The simplest form is password authentication, where the challenge is asking for the password and the password itself is the only valid response. However, more complex challenge-response authentication mechanisms are especially used to avoid that attackers may acquire the

secret knowledge. If, for example, a series of passwords is used, and the device asks for a specific one, say the n-th one, then the interception of a single password will not necessarily put an attacker into the position to authenticate on the device.

Graphical passwords can be used to avoid the use of passwords that are hard to remember. Humans are better in remembering images than in remembering complex character strings. Image-based authentication can be based on recall or based on recognition [2]. Recall-based systems require users to remember images. Recognition-based graphical passwords are based on the identification whether the user has seen an image before, i.e., previously seen images have to be recognized rather than generated from memory. Most mobile devices provide PINs and passwords for authentication by knowledge. Graphical passwords are better suited for touch screens than any textual input. Android also provides the possibility to draw a pattern for authentication.

2.2 Ownership

Smartcards or electronic tokens are examples of "what we have" type of authentication. A simple counterpart example in the non-digital world is a key that opens a door. Whoever is in the possession of the key can enter the door. It is important that there can be copies of the key allowing multiple persons to enter the door. Other digital examples are RFID (Radio-frequency Identification) tags or magnetic stripe cards. NFC (Near Field Communication) and is a standard for radio communication for short distances up to 10 cm [3]. Ownership-based authentication based on NFC tags requires holding the NFC tag close to an NFC reading device. For comparison, the identification number of the NFC tag can be used. Advantages of authentication by ownership include the generation of secure passwords by an owned item. However, care has to be taken to remain in the possession of the item. When lost or stolen, another person may successfully authenticate herself. A big threat is given when a copy of the item is made and goes unnoticed. Authentication by ownership is not used yet for mobile devices.

2.3 Inherence

Biometric-based authentication schemes are examples of the "what we are" type of authentication. Biometric characteristics include finger prints, faces, the iris, voices, the handwriting, the gait, gestures, etc. They can be classified into static and dynamic characteristics. Static methods are based on what a person is, whereas dynamic methods are based on how a person does something. Finger prints are impressions left by friction ridges of fingers [4]. They are a widely used biological recognition technique and have been used for personal verification for many decades. Today, automatic fingerprint recognition is state-of-the-art, enabled by technologies of image processing and pattern recognition. Android provides the only mobile authentication mechanism by inherence, i.e., a face unlock mechanism where the user can authenticate via face recognition.

3. ATTACKS

Authentication is used to secure access to systems and services. Attacks against authentication result in illegitimate users impersonating legitimate users. Thus, attackers can use systems and services and perform activities in the name of the legitimate user. This is a threat against confidentiality, integrity and availability. Subsequently, we will describe various forms of attacks and discuss whether they are an issue in our context.

3.1 Capturing

Capturing attacks mainly involve things that may be captured. Examples are social engineering, shoulder surfing and eaves dropping. Social engineering is the art of manipulating people into performing actions or revealing confidential information [6]. Social engineering is often done in enterprise environments where people, for example, do not personally know all staff members of technical support. Often a phone call may be sufficient to get useful information [7]. Shoulder surfing means watching someone entering sensitive information, often on a smart phone where gestures are used for authentication and can be recognized quite easily. Another form of capturing is done by spyware, i.e., malware that collects information about users without their knowledge. This knowledge may include information about authentication. Authentication on mobile devices is prone to social engineering, shoulder surfing and spyware. Eavesdropping is not an issue yet, but could become a factor when, for example, authenticating wirelessly via an NFC tag.

3.2 Cracking

In contrast to capturing, cracking does not require any interaction with the legitimate user. Cracking involves systematic trying to find out what the system is accepting for successful authentication. Guessing may be successful when users fail in creating a secure password. Insecure passwords are easy to remember, but also easy to guess [8]. Guessing attacks are often combined with social engineering where the attacker is trying to get as much information about a user as possible [9]. Users who choose weak passwords are more prone to attacks than others, because attackers will naturally enter such passwords first. It is recommended to avoid references to personal data when creating a password. Examples are birth dates, place of residence, names of partner or children [5]. Dictionary attacks use a list of often used passwords [10]. Brute-force attacks are similar but use any combination of possible characters [5]. Hybrid attacks combine dictionary attacks with brute-force attacks. Probable lists of passwords are used with systematic modifications of these passwords like appending some characters or switching upper and lower case letters [11]. Dictionary and brute-force attacks are typically done automatically. In principle, they can be performed in our context, but without automation they still remain in the category of guessing.

3.3 False identities

Legitimate users may be misled by attackers pretending false identities. In spoofing attacks some masquerades herself as someone else by falsifying data. Spoofing can take many forms, e.g., e-mail spoofing, IP spoofing, referrer spoofing, website spoofing. Website spoofing, for example, involves the creation of a hoax website that looks identical to the original website. The objective is usually fraudulent, e.g., phishing, which is an attempt to unwarrantedly acquire sensitive information, e.g., credit card details.

Man-in-the-middle attacks are a special form of spoofing attacks. Attackers make independent connections between users and the servers that they believe to be connected with. If attackers are able to intercept all messages between two interconnected parties, they can control the entire conversation and gain access to sensitive information between these parties even when it is encrypted. In our authentication context, phishing may be an issue as a rogue application may fake the authentication screen and lure a user into

disclosing her credentials. Spoofing and man-in-the-middle-attacks are not relevant in the context of mobile devices.

3.4 Physical attacks

Physical attacks include theft and duplicates. Theft and duplications mainly involve things that we own. Attackers can steal a smart card but not a password as long as it is only in our head. As soon as we write it down, say on a sheet of paper, it may get stolen. This may even happen without being noticed by ourselves. The same is true for duplicates. Dumpster diving is also in this category, because attackers may, for example, find the sheet of paper with the password on it in our trash can. Hardware manipulation is another form of physical attacks. It may yield in duplication. A typical example is an ATM skimmer where the attacker adds hardware to the regular ATM that allows her to make a copy of the ATM card and to get an eye on the PIN entering of the legitimate user [12]. Hardware manipulation will hardly go unnoticed with a mobile device, but theft, duplicates and dumpster diving pose a threat in our context.

4. SECURELOCK

The Android application SecureLock provides four different authentication methods as well as some additional functionality. Users can select one or several of these methods for authentication or let the application choose purely by chance. SecureLock is intended for a replacement of Android's lock screen.

4.1 PIN and Password

PINs are frequently used to authenticate to the SIM card of the network operator. Authentication succeeds if the entered sequence matches with the one defined by the network operator, or, as in our case, as defined during the registration phase. Unlike other systems, the length of the PIN in SecureLock is not subject to any restrictions. Similar to PINs, our password authentication does not have any restrictions in its length. The password may contain any upper- and lower-case characters, digits and special characters.

4.2 GesturePuzzle

GesturePuzzle is a knowledge-based authentication method. It combines two different types of graphical passwords. On the one hand a recognition-based graphical password is used where the user has to recognize certain images. On the other hand a pure recall-based graphical password requires the user to reproduce an image, more specifically, a gesture. This gesture has to be entered via the touch screen. Authentication is based on one or several passwords. A password in GesturePuzzle is a quantity of at least one image that results in a specific gesture. Thus, users have to remember one resulting gesture per specified combination of images. An example GesturePuzzle password is shown on the left of Figure 1. The image combination consists of a sunshade, a muffin and a strawberry. This combination results in the square gesture shown below. After recognizing the image combination, this gesture has to be entered by the user for authentication.

To protect against shoulder surfing not only the image combination of a password, but a matrix of an arbitrary number of images is presented for authentication. The user does not have to consider all images of this matrix. She only has to check a small section which had been defined during setup. This relevant area provides the advantage that an attacker cannot determine the image combination without additional knowledge about the relevant area. For authentication, the corresponding gesture must be drawn. The right side of Figure 1 shows an example of a screen for authentication with GesturePuzzle. The relevant area is in the upper left corner. As it contains a sunshade, a muffin and a strawberry, the square gesture has to be drawn. For a better understanding the relevant area is shown with a different background color in Figure 1. The area is not visually recognizable when in real use.

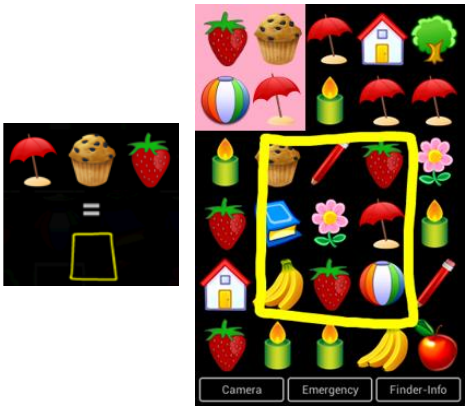

Figure 1. Authentication in GesturePuzzle

All images in the matrix are randomly placed and change with each authentication. The relevant area contains always exactly one of the specified passwords. The relevant area in the example of Figure 1 contains the password shown on the left. Therefore, the user has to enter the square gesture at any point on the screen to get authenticated. During password registration, the size of the image matrix and the relevant area can be defined arbitrarily. It is also possible to decide on gesture accuracy, i.e., how accurately a gesture at least has to be reproduced for successful authentication. Additionally, an arbitrary number of passwords can be defined. It is also possible to decide whether the trace of the entered gesture will be displayed on the screen. Not showing the trace provides additional protection against shoulder-surfing attacks, because it is more difficult to spy out the exact gesture.

4.3 NFC

SecureLock offers an ownership-based authentication method based on NFC tags. Multiple tags can be registered. Needless to say, NFC authentication is offered only to users with enabled NFC hardware. For comparison, the identification number of the NFC tag is used.

4.4 Miscellaneous

SecureLock offers a series of additional functions like the camera feature which enables users to take pictures even without prior authentication. This is useful for snapshots where users may lose too much time when having to authenticate first. Without authentication, users are only allowed to take pictures. Pictures taken previously cannot be viewed. Finder information can be specified as sort of a business card that is accessible to finders if the phone gets lost. If a device is equipped with a front camera, SecureLock optionally takes a picture when authentication fails. Thus, the legitimate user can later have a look at images of people who tried to authenticate on her device without success. In some scenarios a period of validity is useful. Within this period, the user does not have to re-authenticate. This function is beneficial when a device is frequently used or not prone to get stolen or lost, e.g., during a long drive with the car. As a fallback strategy a PUK, i.e., a per-

sonal unblocking key has been implemented. The PUK can be used if authentication data such as a password or gestures get forgotten, or if required NFC tags get stolen or lost. The PUK is similar to a password and consists of an arbitrary string.

4.5 Implementation Aspects

The aim of SecureLock is to deny the usage of the device without prior authentication. Thus, SecureLock is intended as a substitute for Android's screen lock. Android does not provide an API for screen lock replacements. Therefore, it is necessary to implement the application as an Android home screen. When a user presses the device's home button, she will directly be passed on to the home screen. This would bypass an authentication application. However, Android allows an application to be defined as the home screen. Therefore, the user will end up in SecureLock after pushing the home button. SecureLock then will authenticate the user and redirect her to Android's standard home screen. Authentication relevant data is stored with AES encryption and is kept in those storages that cannot be accessed by other applications.

4.6 Installation

The Android application SecureLock can be installed via Google Play on Android devices with version 2.3.3 or higher. Thus, currently SecureLock can be installed on 82.9% of all Android devices [13]. Because SecureLock has to be used as home screen, it is necessary after installation that users define it as the standard home screen. This can be done via the options menu that appears after the first use of the home button. As mentioned previously, SecureLock is meant to be a replacement of the Android screen lock. Therefore, the original Android lock should be inactive, because otherwise the user would have to authenticate twice in two different systems. When an Android screen lock like the unlock pattern, Face Unlock, PIN or password is set, then authentication via SecureLock will appear first. Once authenticated to SecureLock, the user will additionally have to authenticate to the operating system.

5. COMPARISON AND EVALUATION

We have evaluated SecureLock in two steps. First, we have compared the authentication methods PIN, password, unlock pattern, Face Unlock, GesturePuzzle and NFC. If GesturePuzzle is used with only one gesture, than it is comparable to the unlock pattern. Therefore, for comparison we assume that more than one gesture is used. Also, SecureLock provides many options and, for example, allows users to use a PIN or password only. Therefore, we have chosen to evaluate SecureLock when used with a GesturePuzzle with several passwords combined with an NFC tag.

Table 1. Comparison of Lock Screen Authentication

	PIN	Password	Unlock Pattern	Face Unlock	GesturePuzzle [1]	NFC Tags	SecureLock [2]
Knowledge	●	●	●		●		●
Ownership						●	●
Inherence				●			

[1] With more than one password
[2] Combination of GesturePuzzle with more than one password and NFC tag

Table 1 categorizes these mechanisms according to the authentication categories described in Section 2. Please note that SecureLock had been designed to be extensible. Thus, the integration of an authentication method by inherence as Face Unlock would be easy. However, Face Unlock does not provide an API yet. Therefore, its integration will require extensions in Android.

5.1 Related Work

Android, BlackBerry OS, iOS, Symbian and Windows Phone provide locking mechanisms. The use of passwords is available on all platforms. There are no restrictions to passwords, such as minimum length or use of special characters. Authentication with PIN is supported by Android and iOS. Android allows PINs with 4-17 digits, while iOS requires exactly 4 digits. Android's unlock pattern is a graphical password, in which a path is drawn on a matrix with 3x3 points. The pattern must comprise at least four points of the matrix. Each of the nine points may be used not more than once. Android has also introduced *Face Unlock* to authenticate via face recognition. This feature requires a front camera at which a user simply has to look at. Related work is confined to solutions provided by operating systems for mobile devices. Application developers are not supposed to provide their own lock screens for these devices. Thus, it is hardly surprising that there are no programming interfaces that allow for the implementation and integration of such a feature. SecureLock provides an alternative mechanism but, as mentioned in Section 4.6, some extra steps during installation have to be taken that are demanding for inexperienced users.

5.2 Security

In Section 3 we have discussed several attacks. Some of them are not an issue in our context as they aim for information sent over the Internet. We will consider attacks that are relevant when authenticating on a mobile device.

Social engineering includes the manipulation of people to reveal confidential information like a PIN or password. It is rather easy to communicate PIN, password or an unlock pattern. Social engineering may also result in the exposure of an image that could be used for Face Unlock. NFC tags are also at risk, because an attacker may come close enough to read the tag. We assume GesturePuzzle to be resistant to social engineering, because several passwords are used and for each password a series of images plus the corresponding gesture would have to be revealed.

Shoulder surfing. If attackers watch someone who is authenticating on her mobile device, they may easily recognize a PIN or an unlock pattern. A password is somewhat harder to spy out due to its length. Unlock patterns are particularly vulnerable to shoulder surfing, because they are drawn on the screen and can therefore be spied out even from a distance. All the other mechanisms do not pose a threat when being watched by others.

Malware can take multiple forms. For example, fake applications or spyware can sit in the background, log user input and send it to a server controlled by the attacker. Once authentication data is on the server and, thus, in the hands of the attacker, she may try to get physical access to the device. It is easy to log a user's input when entering a PIN or a password. The same holds for an unlock pattern. Even Face Unlock and NFC authentication can be at stake. The only information needed is an image or the identification number of the NFC tag. The screen dialog to query a PIN

requires only a few graphical elements. Fake applications with the exact same graphical user interface may be designed to ask users for their PIN or password. GesturePuzzle provides some protection because input depends on images shown to users.

Guessing is a promising attack if a mobile device is in the hands of an attacker, unless the device deactivates itself for increasing intervals when wrong guesses are entered. Face Unlock, NFC tags and SecureLock do not permit any forms of guessing. Unlock patterns may pose a threat as always the same pattern is entered. Fingers leave greasy residue on the touch screen. It is possible to analyze traces on the screen and perhaps deduce the pattern [14].

Duplicates are an issue for Face Unlock and for NFC tags. Copies of an NFC tag can be made. There are reports in user forums that Face Unlock can be bypassed by using a photo of the legitimate user. **Theft** is only an issue for authentication by ownership, thus for NFC tags. But Face Unlock is also at risk as a photo may get stolen and later used to authenticate on the device.

Dumpster diving is an issue if users write down their authentication credentials and dispose of their notes. Attackers may get hold of any information. We assume that NFC tags are spared from such attacks as it is unlikely that users throw away their tags when still used for authentication.

Unawareness of users is a security risk in many situations. Several users consider it superfluous to protect their devices with a lock screen. They deem their device secure as it is in their possession and as they carry it along all the time. They do not waste much thought about their device getting lost or stolen. Unawareness is also a problem when users do authenticate on their device. They may use a weak PIN or password that is prone to guessing or shoulder surfing. Unlock patterns and GesturePuzzle may also suffer patterns that are not carefully chosen. Face Unlock, NFC tags and SecureLock have fewer problems with unaware users.

Table 2 summarizes our perceptions of security aspects. Check marks indicate that security is sufficient. A dash has the opposite meaning. Circles indicate some degree of security that we consider to be insufficient. As shown in the table, SecureLock is more secure than other mechanisms. This security advantage is attributed to the combination of two different authentication mechanisms.

Table 2. Comparison of Security Aspects

	PIN	Password	Unlock Pattern	Face Unlock	GesturePuzzle	NFC Tags	SecureLock
Social engineering	-	-	-	o	✓	o	✓
Shoulder surfing	-	o	-	✓	✓	✓	✓
Malware	-	-	-	-	✓	-	✓
Guessing	o	o	o	✓	o	✓	✓
Duplicates	✓	✓	✓	-	✓	-	✓
Theft	✓	✓	✓	-	✓	-	✓
Dumpster Diving	-	-	o	-	o	✓	✓
Unawareness	-	-	o	✓	o	✓	✓

5.3 Usability

The usability of modern mobile devices is influenced by the use of touch screens and the duration of the time it takes to unlock the device. Additionally, we evaluate the complexity, i.e., how much users have to remember in order to successfully authenticate, as well as the reliability of the system.

Touch screen. The usability of PINs and especially passwords is limited in the context of mobile devices, mainly because mobile devices are typically equipped with a touch screen rather than a hardware keyboard like traditional computers. Virtual keyboards are inconvenient to enter secure passwords, which ideally contain a variation of uppercase and lowercase letters, digits and special characters [15]. The Android unlock pattern, GesturePuzzle and, thus, SecureLock are better suited for authentication via touch screen. NFC tags are independent from screen and keyboard.

Duration. The duration of the authentication process is crucial for user acceptance. We roughly estimate 4 sec to enter a PIN and 10 sec to enter an average password. The unlock pattern and Face Unlock take less than a PIN. The Android unlock pattern is only secure if it uses a long path. But the longer the pattern, the more time is needed for authentication, again resulting in reduced usability. Therefore, most users prefer rather short patterns, so that the authentication can be carried out within a short period of time. GesturePuzzle takes a little longer than the unlock pattern because users have to analyze the images in the relevant area. Use of an NFC tag will take 2-5 sec, depending on where the tag is carried and how easily accessible it is. As SecureLock combines GesturePuzzle and NFC tags it will take a little longer. We estimate 5-8 sec which is less than the input of an average password.

Complexity. Images are typically much better to remember for humans than text, unless the text is short like a four-digit PIN. Unlock patterns may get quite complex unless it is a simple and insecure circle or square. The same holds for GesturePuzzle with the additional burden that more than one pattern has to be remembered in addition to sets of images.

Reliability. Android´s Face Unlock is promising but still struggling with usability problems. For example, faces of people with dark skin are not always recognized. For authentication, the front camera of the smart phone, which has no flash light, is used. Therefore, faces in low light are not correctly recognized.

Table 3 summarizes our usability perceptions. Expectedly, PINs and NFC tags do quite well. SecureLock's rating has circles for duration and complexity, the price that had to be paid for security. NFC tags do best in usability, closely followed by unlock patterns.

Table 3. Comparison of Usability Aspects

	PIN	Password	Unlock Pattern	Face Unlock	GesturePuzzle	NFC Tags	SecureLock
Touch Screen	o	-	✓	✓	✓	✓	✓
Duration	✓	-	✓	✓	✓	✓	o
Complexity	✓	-	o	✓	o	✓	o
Reliability	✓	✓	✓	-	✓	✓	✓

5.4 Survey

We have additionally evaluated the usability of SecureLock by a small survey (N=11). Due to the small sample size the results of the survey can be regarded only as a first impression and as a tendency. The evaluation has shown that authentication on mobile devices, especially on smart phones, had been seen as important. SecureLock has been assessed with high usability with space for further improvements. GesturePuzzle was preferred by most participants because it allowed for a rapid authentication and was easy to use on touch screens. Several users noted that the Android unlock pattern was considered sufficient to meet their security needs. Random authentication was least frequently used. This was justified by the fact that this was complicated because much knowledge had to be kept in mind. Some users argued that they would neither use PIN nor password, as they require too much time for authentication. SecureLock is classified by the evaluation participants as a secure authentication application. However, we have to state that security can be further enhanced by a direct integration into the Android operating system.

5.5 Additional Features

Additional features are independent from authentication mechanisms, yet have an influence on the usability and users' willingness to accept a specific mechanism. Thus, features like pictures of users who authenticate without success have to be compared with lock screen solutions of iOS or Android. For example, iOS also offers access to the device's camera without prior authentication. Android offers a period of validity as well as finder information. In our survey, the finder information, the attacker's photograph and the camera function for snapshots were most popular.

6. CONCLUSION

We have introduced authentication mechanisms by knowledge, by ownership, and by inherence. PINs and gestures are the most user-friendly and, thus, the most frequently used authentication mechanisms for mobile devices. Once lost or stolen, mobile devices pose a serious threat to both sensitive data on the device and sensitive data accessed via the device. We have suggested a combined authentication mechanism for Android where users can choose among various different mechanisms and can also use a random selection of these mechanisms. An evaluation of our approach has shown that many users still are not aware of the perils in using mobile devices and that they favor more usable approaches to more secure ones. We believe that operating systems of mobile devices have to provide more advanced authentication mechanisms and additional features as demonstrated by our approach, e.g., images of attackers or e-mail messages upon wrong authentication attempts. Additionally, we imagine an approach where authentication varies depending on parameters like period of inactivity or last used application. Simple and more complex authentication mechanisms could also vary depending on the sensitivity of used applications or depending on geographic location, requiring simpler authentication if on familiar terrain.

7. REFERENCES

[1] Almuairfi, S., Veeraraghavan, P., and Chilamkurti, N. 2011. IPAS: Implicit Password Authentication System. In *IEEE Workshops of International Conference on Advanced Information Networking and Applications Workshops* (Biopolis, Singapur, March 22 - 25, 2011). WAINA'11. IEEE, Piscataway, NJ, 430–435.

[2] Khan, W. Z., Xiang, Y., Aalsalem, M. Y., and Arshad, Q. 2011. A Hybrid Graphical Password Based System. In *11th International Conference on Algorithms and Architectures for Parallel Processing* (Melbourne, Australia, Oct. 24 - 26, 2011), *ICA300'11*. Springer, Berlin, Heidelberg, 153–164.

[3] Chavira, G., Nava, S. W., Herváz, R., Villarreal, V., Bravo, J., Martín, S., and Castro, M. 2008. Services through NFC technology in AmI environment. In *Proceedings of the 10th International Conference on Information Integration and Web-based Applications & Services* (Linz, Austria, November 24 - 26, 2008). iiWAS '08. ACM, NY, 666–669.

[4] Maltoni, D. and Cappelli, R. 2008. Fingerprint Recognition. In *Handbook of Biometrics*, A. K. Jain, P. J. Flynn and A. A. Ross, Eds. Springer, New York, NY, 23–42.

[5] Clarke, N. 2011. *Transparent User Authentication*. *Biometrics, RFID and Behavioral Profiling*. Springer, London, NY.

[6] Goodchild, J. 2012. *Social Engineering: The Basics*. http://www.csoonline.com/article/514063/social-engineering-the-basics. Accessed 23 July 2012.

[7] Orgill, G. L., Romney, G. W., Bailey, M. G., and Orgill, P. M. 2004. The Urgency for Effective User Privacy-education to Counter Social Engineering Attacks on Secure Computer Systems. In *Proceedings of the 5th conference on Information technology education* (Salt Lake City, Utah, October, 28 – 30, 2004). SIGITE '04. ACM, New York, NY, 177–181.

[8] Gong, L., Lomas, M. A., Needham, R. M., and Saltzer, J. H. 1993. Protecting Poorly Chosen Secrets from Guessing Attacks. *IEEE Journal on Selected Areas in Communications* 11, 5 (Jun. 1993), 648–656.

[9] Barber, R. 2001. Social engineering: A People Problem? *Network Security* 2001, 7 (Jul. 2001), 9–11.

[10] Adams, C. 2011. Dictionary Attack. In *Encyclopedia of Cryptography and Security*, H. C. A. van Tilborg and S. Jajodia, Eds. Springer, New York, Dordrecht, Heidelberg, London, 332-332.

[11] Vu, K.-P. L., Proctor, R. W., Bhargav-Spantzel, A., Tai, B.-L., Cook, J., and Schultz, E. E. 2007. Improving password security and memorability to protect personal and organizational information. *International Journal of Human-Computer Studies* 65, 8 (Aug. 2007), 744–757.

[12] Federal Bureau of Investigation 2011. *Taking a Trip to the ATM? Beware of 'Skimmers'*. http://www.fbi.gov/news/stories/2011/july/atm_071411. Accessed 23 July 2012.

[13] Android Developers 2012. *Platform Versions*. http://developer.android.com/resources/dashboard/platform-versions.html. Accessed 13 October 2012.

[14] Hoog, A. 2011. *Android Forensics*. *Investigation, Analysis and Mobile Security for Google Android*. Elsevier, Waltham, MA.

[15] Flick, T. and Morehouse, J. 2011. *Securing the Smart Grid: Next Generation Power Grid Security*. Elsevier Science, Burlington, MA.